8-2020

Context Sensitive Image Denoising and Enhancement using U-Nets

Sahaj Tushar Gandhi
sxg2475@rit.edu

Context Sensitive Image Denoising and Enhancement using U-Nets

APPROVED BY

SUPERVISING COMMITTEE:

Dr. Thomas B. Kinsman, Supervisor

Dr. Alexander G. Ororbia II, Reader

Dr. Joe Geigel, Observer

Context Sensitive Image Denoising and Enhancement using U-Nets

by

Sahaj Tushar Gandhi

THESIS

Presented to the Faculty of the Department of Computer Science

Golisano College of Computer and Information Sciences

Rochester Institute of Technology

in Partial Fulfillment

of the Requirements

for the Degree of

Master of Science in Computer Science

Rochester Institute of Technology

August 2020

Dedicated to my parents.

Acknowledgments

This has been one adventurous ride and I have shared this experience with a lot of people. I am very happy to have been around these people and I would like to take a moment to thank them.

First, I would like to extend my deepest gratitude to my advisor Dr. Thomas B. Kinsman for having encouraged and guided me through this entire process. I am extremely grateful to have gotten a chance to understand the process of academic research. I would also like to thank Dr. Alexander G. Ororbia II and Dr. Joe Geigel for serving on my committee and providing invaluable and constructive feedback. I would also like to thank Sam Waters for helping me out with resources in the Computer Science department at Rochester Institute of Technology.

I am extremely grateful to Behrooz Mansouri, Nishant Keni, Abishai Dmello, and Paren Desai for all the suggestions, feedback and constant discussions during the process of this thesis.

Last, I would like to thank my parents and family who have been my support-system and who have showered me with unconditional love and guidance in life.

Abstract

Context Sensitive Image Denoising and Enhancement using U-Nets

Sahaj Tushar Gandhi, M.S.
Rochester Institute of Technology, 2020

Supervisor: Dr. Thomas B. Kinsman

Noise in images gets introduced at almost every stage of the camera image signal processing pipeline (ISP). Camera companies provide software that cleans most of the noise added at each stage. Even after noise removal is done by the camera software, different noise patterns with different intensities remain in the image. With advances in deep learning, the algorithms are architectured end-to-end. In the present time, machine learning and deep learning models work as end-to-end systems with a special-purpose feature extraction phase. This thesis focuses on the removal of any residual noise in images as performed during the feature extraction stages. The feature extraction process is done by using the classic segmentation architecture, U-Nets. Traditionally, segmentation models have helped with identifying the locations of objects in images. In this thesis, a U-Net based architecture has been used to identify

important regions in an image in order to localize background noise. With the removal of this noise, the resulting images created are cleaner and provide better content for other tasks like Image Classification, Object Segmentation, and Scene Understanding. MNIST and Fashion-MNIST datasets were used to train the prototypes of the proposed architectures. To build an effective system, a noise model was created to reflect the properties of true noise found in images. Various Gaussian and Speckle noise models were used during the initial prototyping phase, and for the final prototype, a combination of the noise models was used. This combination of noise models represents the true occurrences of noise in images found in nature. Due to the occurrences of multiple types of noise in images, modeling a realistic representation of this noise was done using a Mixture of Gaussians and tested on a complex dataset, ImageNet. The proposed system worked well in denoising complex invisible noise, like adversarial noise, from these images. The effectiveness of the proposed approach was evaluated using signal-structure metrics such as PSNR and SSIM, along with metrics such as Precision, Recall, and F1-score that are used to quantify the improvements made during computer vision tasks.

Table of Contents

Acknowledgments **v**

Abstract **vi**

List of Tables **x**

List of Figures **xi**

Chapter 1. Introduction **1**

Chapter 2. Background **4**

2.1 Image Processing Techniques 5
2.2 Deep Learning Models 10
2.3 Segmentation Algorithms 16
2.4 Noise Models 17
2.5 Evaluation Techniques 19
2.6 Hyperparameters 21
2.7 Skip Connections 22
2.8 Loss Functions 23

Chapter 3. Methodology **26**

3.1 Hypothesis. 26
3.2 Approach 28
3.3 Architecture 29

Chapter 4. Experiments **35**

4.1 Types of Models 35
4.2 Types of Datasets 39
4.3 Image Sizes 42

4.4 Noise Types 44
4.5 Types of Losses 47
4.6 Optimizer Types 47
4.7 Skip Connections 49

Chapter 5. Results and Discussion **52**
5.1 Evaluation Process 52
5.2 Results 58

Chapter 6. Future Work **94**

Chapter 7. Conclusion **96**

Bibliography **98**

List of Tables

5.1 Performance of the Baseline Architectures and the two proposed Architectures, when Skip Connections were not a part of the Outer Encoder-Decoder Architecture. 62

5.2 Performance of the Baseline Architectures and the two proposed Architectures, when Skip Connections were added as a part of the Outer Encoder-Decoder Architecture. 62

5.3 Metric scores of the effectiveness of the original test images on the pretrained model on the FashionMNIST dataset. 74

5.4 Metric scores of the effectiveness of the enhanced test images on the pretrained model on the FashionMNIST dataset after being passed through the proposed system. 75

5.5 PSNR and SSIM metric scores for models trained with noise modeled as a Mixture of Gaussians on the MNIST and Fashion-MNIST datasets respectively, when tested on images from the ImageNet dataset. 82

List of Figures

1.1 Gaussian, local Gaussian, Poisson, Salt and Pepper, and Speckle Noise when added to the image labeled "No Noise". 2

2.1 Image depicting the four Kuwahara quadrants [27], to calculate the pixel value for the central image, that are used to calculate pixel values by using the mean value of the quadrant having the smallest variance. 7

2.2 Image depicting the nine sub-windows, used by the Matsuyama and Nagao filter [39], that are used to calculate pixel values by using the mean of the sub-window with the lowest variance. . 8

2.3 Flow of information during the forward propagation process in a deep neural network. 11

2.4 Flow of information during the backward propagation process in a deep neural network. 12

2.5 Generic Representation of an Autoencoder. 14

2.6 Generic Representation of a Convolutional Neural Network. . . 15

2.7 Generic Representation of a Generative Adversarial Network. . 16

3.1 Inner U-Net Model [46]. 29

3.2 Baseline Encoder-Decoder Architecture without the inner segmentation U-Net model. 29

3.3 Proposed Architecture that uses the output of the inner segmentation U-Net algorithm, the Saliency Map, as input to the outer Encoder-Decoder Architecture. 30

3.4 Proposed Architecture that uses the output of the inner segmentation U-Net algorithm, the Saliency Map, along with the Noisy Image added channel-wise as input to the outer Encoder-Decoder Architecture. 30

3.5 Example of image transformation at various stages of the proposed model, that uses the output of the U-Net model, the Saliency Map, as input to the outer Encoder-Decoder model. . 32

3.6 Example of image transformation at various stages of the proposed model, that uses the output of the U-Net model, the Saliency Map, along with the original noisy image appended channel-wise, as input to the outer Encoder-Decoder model. . 33

4.1 The Saliency Map of an image, depicting the number 7, taken from the MNIST dataset, after being processed through the U-Net segmentation model. 36

4.2 The Original image, depicting the number 7, taken from the MNIST dataset, before being processed through the proposed architectures. 36

4.3 The Saliency Map of an image, depicting a pair of trousers, taken from the Fashion-MNIST dataset, after being processed through the U-Net segmentation model. 37

4.4 The Original image, depicting a pair of trousers, taken from the Fashion-MNIST dataset, before being processed through the proposed architectures. 38

4.5 Examples of images taken from the MNIST dataset [29]. . . . 40

4.6 Examples of images taken from the Fashion-MNIST dataset [57]. 41

4.7 Examples of images of dogs taken from the ImageNet dataset [8]. 42

4.8 Effect of Gaussian, Local Gaussian, Poisson, Salt and Pepper, and Speckle Noise, when added to an image. This is a horizontal representation of Figure 1.1. 44

4.9 Effect of different type of noise on the original image of a number 7, taken from the MNIST dataset. Gaussian noise, Speckle noise, a combination of Gaussian and Speckle noise, noise modeled as a Mixture of Gaussians and Adversarial (FGSM) noise added to the original image. 46

4.10 Loss curve of the Baseline Encoder-Decoder Model when it is trained with the Adam optimizer. 48

4.11 Loss curve of the Baseline Encoder-Decoder Model when it is trained with the Stochastic Gradient Descent optimizer. 49

4.12 Loss curve of the Baseline Encoder-Decoder Model when skip connections were added to the architecture. 50

4.13 Loss curve of the Baseline Encoder-Decoder Model when skip connections were not a part of the architecture. 51

5.1 Instances where the proposed model reconstructed a blank image, when training on the MNIST dataset. Here, the top image is the noisy image, middle image is the expected denoised image as an output from the proposed model, and the bottom image is the original image. 60

5.2 Instances where the proposed model reconstructed a blank image, when training on the Fashion-MNIST dataset. Here, the top image is the noisy image, middle image is the expected denoised image as an output from the model, and the bottom image is the original image. 61

5.3 Instance of an Image from the MNIST [29] dataset, depicting number 9. 65

5.4 Instance of an Image from the Fashion-MNIST [57] dataset, depicting a pullover. 66

5.5 Instance of an Image from the MNIST [29] dataset, depicting number 9, as seen in Figure 5.3, with noise added to the image. 66

5.6 Instance of an Image from the Fashion-MNIST [57] dataset, depicting a pullover, as seen in Figure 5.4, with noise added to the image. 67

5.7 Saliency Map generated for the image as seen in Figure 5.3, taken from the MNIST [29] dataset, when it is passed through the inner U-Net segmentation model. Here, we can see the noise patterns mapped in terms of the context of the image, in this case, the number 9. 67

5.8 Saliency Map generated for the image as seen in Figure 5.4, taken from the Fashion-MNIST [57] dataset, when it is passed through the inner U-Net segmentation model. Here, we can see the noise patterns mapped in terms of the context of the image, in this case, a pullover. 68

5.9 The denoised and enhanced version of the noisy Image as seen in Figure 5.5, taken from the MNIST [29] dataset, when it is passed through a proposed architecture. Here, we can see how well-defined the image context is along with the extent to which denoising was done. 69

5.10 The denoised and enhanced version of the noisy Image as seen in Figure 5.6, taken from the Fashion-MNIST [57] dataset, when it is passed through a proposed architecture. Here, we can see how well-defined the image context is along with the extent to which denoising was done. 70

5.11 A combined version of the image from Figure 5.3, taken from the MNIST [29] dataset, that depicts the image at three different stages of the model, where the topmost image is the original noisy image, the image in between is the denoised and enhanced image and the image at the bottom is the original image before noise was added to it. 71

5.12 A combined version of the image from Figure 5.3, taken from the Fashion-MNIST [57] dataset, that depicts the image at three different stages of the model, where the topmost image is the original noisy image, the image in between is the denoised and enhanced image and the image at the bottom is the original image before noise was added to it. 73

5.13 Instance of an Image from the MNIST [29] dataset, depicting the number 7, with noise modeled as a Mixture of Gaussians is added to the image. 76

5.14 Instance of an Image from the Fashion-MNIST [57] dataset, depicting a shoe, with noise modeled as a Mixture of Gaussians is added to the image. 76

5.15 The original image and the corresponding Saliency Map generated from the inner segmentation U-Net model, taken from the MNIST dataset [29] to which we added a combination Speckle and Gaussian noise. 78

5.16 An image with the noisy image, denoised and enhanced image and original image stacked in that order for a clear representation of the image during the process, when a combination Speckle and Gaussian noise was added to an image from the MNIST dataset [29]. 79

5.17 The original image and the corresponding Saliency Map generated from the inner segmentation U-Net model, taken from the Fashion-MNIST dataset [57] to which we added a combination Speckle and Gaussian noise. 80

5.18 An image with the noisy image, denoised and enhanced image and original image stacked in that order for a clear representation of the image during the process, when a combination Speckle and Gaussian noise was added to an image from the Fashion-MNIST dataset [57]. 81

5.19 The Accuracy-Epsilon plot for values of epsilon ranging from 0 to 0.3, where the accuracy is of a classification model on the MNIST dataset [29]. 85

5.20 The Accuracy-Epsilon plot for values of epsilon ranging from 0 to 0.3, where the accuracy is of a classification model on the Fashion-MNIST dataset [57]. 86

5.21 Examples of images from the test set that were affected by FGSM noise and classified by a pretrained LeNet model. This image depicts the image being tested, with its ground truth and predicted values displayed above the image, in that order respectively. The images were taken from the MNIST dataset [29]. 87

5.22 Examples of images from the test set that were affected by FGSM noise, passed through the proposed system, and then classified by a pretrained LeNet model. This image depicts the image being tested, with its ground truth and predicted values displayed above the image, in that order respectively. The images were taken from the MNIST dataset [29]. 88

5.23 Examples of images from the test set that were affected by FGSM noise and classified by a pretrained ResNet model. This image depicts the image being tested, with its ground truth and predicted values displayed above the image, in that order respectively. The images were taken from the Fashion-MNIST dataset [57]. 89

5.24 Examples of images from the test set that were affected by FGSM noise, passed through the proposed system, and then classified by a pretrained ResNet model. This image depicts the image being tested, with its ground truth and predicted values displayed above the image, in that order respectively. The images were taken from the Fashion-MNIST dataset [57]. 90

5.25 An image with the noisy image, denoised and enhanced image, and original image stacked in that order for a clear representation of the image during the process of denoising, when a Adversarial (FGSM) perturbations was added to the image, taken from the MNIST dataset [29]. 92

5.26 An image with the noisy image, denoised and enhanced image, and original image stacked in that order for a clear representation of the image during the process of denoising, when a Adversarial (FGSM) perturbations was added to the image, taken from the Fashion-MNIST dataset [57]. 93

Chapter 1

Introduction

The problem of noise in images is well-known. Noise is predominantly present in some form or the other in every image. Noisy images are problematic to deal with, and often get dropped from datasets when they cause the model to learn the wrong features. Images with lower resolution do not hold enough context and information, due to small number of pixels in the image. Different types of noise get introduced during various stages in the image processing pipeline. Identifying and using the noise artifacts from the images to enhance the images, can help provide better looking images.Denoising would also improve details in the image which can help machine learning models learn better features. This improvement in the current datasets can help improve the accuracy and working of current state-of-the-art algorithms. In the case of sensor noise, denoising and enhancing images can be a software solution to a hardware problem. With a denoising and enhancement algorithm in place, we can make full use of image details. Traditionally, tackling various types of noise in images has been empirically determined. Depending on the type of noise found in an image perceived by the user of the image, steps for noise removal were taken accordingly.

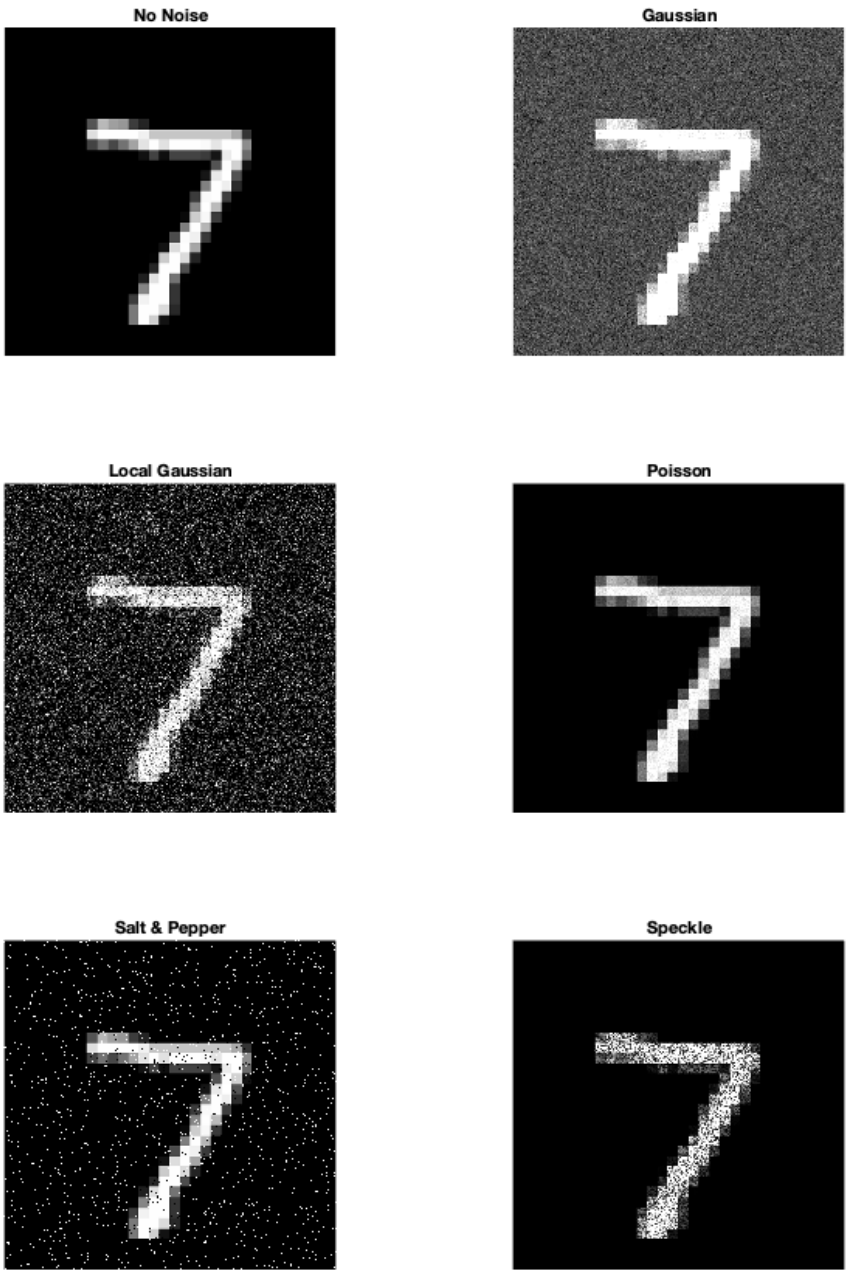

Figure 1.1: Gaussian, local Gaussian, Poisson, Salt and Pepper, and Speckle Noise when added to the image labeled "No Noise".

On identifying noise empirically, filters with various intensities are used to remove maximum visible noise that was identified. Figure 1.1 depicts multiple types of noise and the patterns in images when synthetically added. Here, a simple smoothing function like Gaussian smoothing can be used to clean speckle noise, or median filter can be used to eliminate Salt and Pepper noise.

With machine learning and deep learning being used to analyze large amounts of information, we use one such deep learning algorithm to analyze and understand the pattern of noise occurrence in images. Segmentation algorithms are generally used to identify regions of importance in an image, which helps speed up the process of identifying the object in those regions.

U-Nets optimally analyze and identify important regions in an image based on the task at hand. U-Nets also provide a structure entity, skip connections, that provide an uninterrupted gradient flow while preventing the vanishing gradients problem. Unlike other segmentation algorithms, they are not dependent on a predefined mask during the model's training phase. With that in mind, the Saliency Map they generate would vary with every task. We utilized these advantages provided by U-Nets. They analyze the context of an image for better and quicker prediction. We use the U-Net algorithm to understand the context of an image while also analyzing the pattern of noise in the image.

Chapter 2

Background

Here we create a system for the automatic context sensitive noise removal from images, which would lead to improvements in the downstream processing of computer vision tasks such as object segmentation and scene understanding. By enhancing the images, we are providing more relevant information and context for the machine learning models. The proposed system would be beneficial in cases where the system needs better context information for feature extraction.

Image denoising, in the past, has been performed using heuristic approaches. Depending on the camera, file format, camera ISPs, types of sensors and application type, different types images were affected by different types of noise. Even things like environment, lighting, and other objects in the image would determine the kind of noise which was predominantly present in the images. The cameras and camera software were not as advanced as they are today and as a result, the images were empirically denoised for noise artifacts present in the image, based on the task at hand.

2.1 Image Processing Techniques

When dealing with the removal of Gaussian noise from images, Mean (Averaging) Filter is used to perform this function. The Mean filter is a linear filter, meaning it can be performed using convolution and Fourier multiplication. An undesirable outcome of this filter is the blurring of fine-scaled image edges and details because they also correspond to blocked high frequencies.

Often, image denoising is done using low pass filtering and high pass filtering. With the use of low pass filters in image processing, one can reduce high frequency noise. With this type of noise removal, the image would get smoothed leading to the loss of noise like speckle noise. High pass filtering, on the other hand, are used to make images sharper. While these two types of filters remove noise artifacts in the frequency domain, the image processing done in the spatial domain involves the use of spatial filters like the median filter, which is generally used to remove salt and pepper noise. Median Filter is a non-linear filter. Non-Linear filters are those filters that cannot be performed with convolution or Fourier multiplication.

Similarly, Weiner Filter is also a non-linear filter used to clean speckle noise from images. The Weiner Filter is a pixel-wise adaptive filter and it tailors itself to the local image variance, where little smoothing is performed when variance is large and vice-versa.

As the type of image changes, the type of noise artifacts change. Since hyperspectral images are modeled as a 3-dimensional tensor with two dimen-

sions for spatial data and one dimension of spectral information, denoising methods would involve using matrix algebra to get the best results. Methods like Multiway Weiner Filter (MWF), PARAFAC (Parallel Factor Analysis) Filter, and a combination of multidimensional wavelet packet transform and Multiway Weiner Filter(MWF) have been found to remove noise from images as suggested in [30]. All of the following methods involve the use of matrix algebra. From this we can understand that by understanding the underlying structure of the data being used, we can try understanding the structure of noise inherently present in these types of images. That would help decide the type of approach we could take to denoise and enhance the image's quality.

When dealing with images, the type of noise artifacts depend on the type of modality in which the images were captured. For instance, the pattern in which noise occurs in medical images like Magnetic Resonance Images(MRI), X-ray and Ultrasounds is different from the pattern of noise in images captured by a camera as studied in [37] and [45]. In medical images, one very important thing that we need to consider is that any denoising technique being used should make sure that there is no loss of important signal information from the image. Thus any filter or method being used on such images needs to understand the underlying structure of the image and then remove the noise in the image.

The Kuwahara filter [27] and Matsuyama and Nagao filter [39] were the first to provide a filter which tried to retain some type of context in the images. The Kuwahara filter [27] split the region around the pixel into four

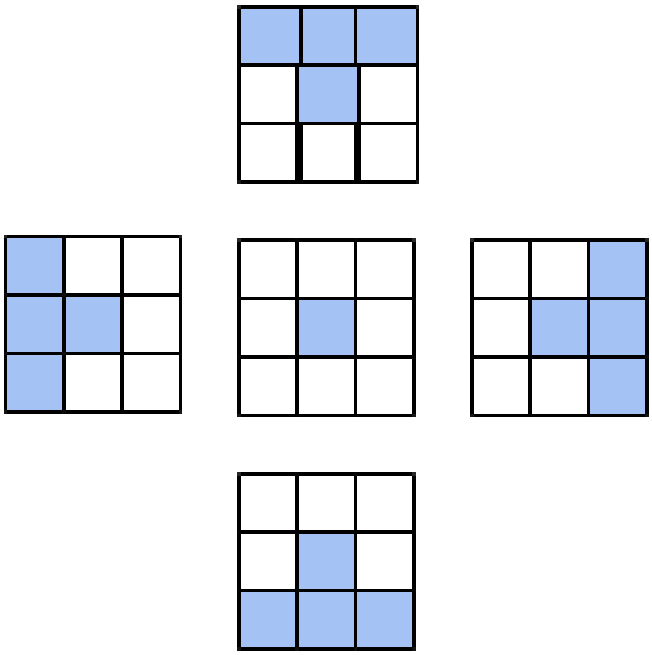

Figure 2.1: Image depicting the four Kuwahara quadrants [27], to calculate the pixel value for the central image, that are used to calculate pixel values by using the mean value of the quadrant having the smallest variance.

quadrants and the variance for each quadrant was calculated. The average of the quadrant having the smallest variance was used to replace the pixel value. Figure 2.1 depicts the four quadrants that the algorithm uses to calculate pixel values.

Nagao et al. [39] made use of nine sub-windows instead of quadrants used by Kuwahara et al. [27]. For each sub-window the variance is calculated and the average of sub-window with the lowest lowest is used to replace pixel

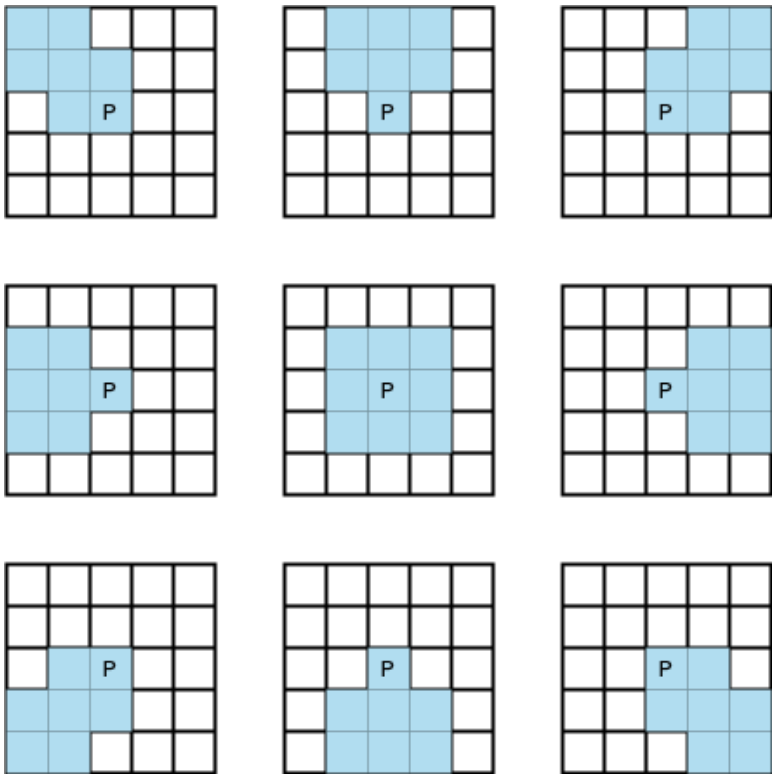

Figure 2.2: Image depicting the nine sub-windows, used by the Matsuyama and Nagao filter [39], that are used to calculate pixel values by using the mean of the sub-window with the lowest variance.

value. Figure 2.2 depicts each of the nine sub-windows used by the Matsuyama and Nagao filter.

Perona and Malik's anisotropic diffusion filter [43] performs context-sensitive image denoising. Their work allowed the removal of noise without affecting the signal in the image by cleaning the images and retaining important image features like lines, edges and textures. The idea was to preserve the image signal while diffusing the noise signal in the image. By doing

so, they perform context-sensitive image denoising. Equation 2.1 defines the anisotropic filter. Here, $c(x, y, t)$ is the diffusion coefficient and $c(x, y, t)$ controls the rate of diffusion and is usually chosen as a function of the image gradient to preserve edges in the image. Pietro Perona and Jitendra Malik pioneered the idea of anisotropic diffusion and proposed two functions for the diffusion coefficient as defined by Equations 2.2 and 2.3.

$$I_t = \frac{dI}{dt} = div(c(x, y, t) \cdot \nabla I) = \nabla c \cdot \nabla I + c(x, y, t)\Delta I \tag{2.1}$$

$$g(||\nabla I||) = e^{-(\frac{||\nabla I||}{k})^2} \tag{2.2}$$

$$g(||\nabla I||) = \frac{1}{1 + (\frac{||\nabla I||}{k})^2} \tag{2.3}$$

Where,

- ∇ is the gradient,
- Δ is the Laplacian,
- $div()$ is the divergence operator, and
- k controls the sensitivity of diffusion near the edges

Bilinear Interpolation and Bicubic Interpolation [16] are a few of the earlier methods used to clean images or produce more information in images

while maintaining context of the image. The process of image denoising and enhancement should retain as much context as it can as it would otherwise lead to the loss of important information in an image. Bilinear Interpolation creates information in a 2D plane while Trilinear Interpolation uses a 3D plane to create information for pixel values. Ragesh et al. [45] also spoke about how there has been a shift from classical image processing methods to more recent machine learning techniques like artificial neural networks and genetic algorithms.

2.2 Deep Learning Models

With more well-defined and complex deep learning models being introduced in the past few years, solving more tasks with deep learning models has started taking place. Deep Learning models are being preferred for solving these various tasks as they provide better generic feature extraction over large datasets. In studies such as [60], [14], and [55], deep learning models like CNNs are used as a special-purpose feature extraction phase to extract optimal features for better task prediction. Another reason why deep learning models are preferred over classical systems is that the parameters used to build optimal classical systems are empirical for every image being used. In contrast, deep learning models can be trained on Graphics Processing Units (GPUs) over entire dataset produce optimal performance.

Due to their capability to generalize on large amounts of data, datasets like ImageNet [8] have been built. This dataset comprises a large amount of

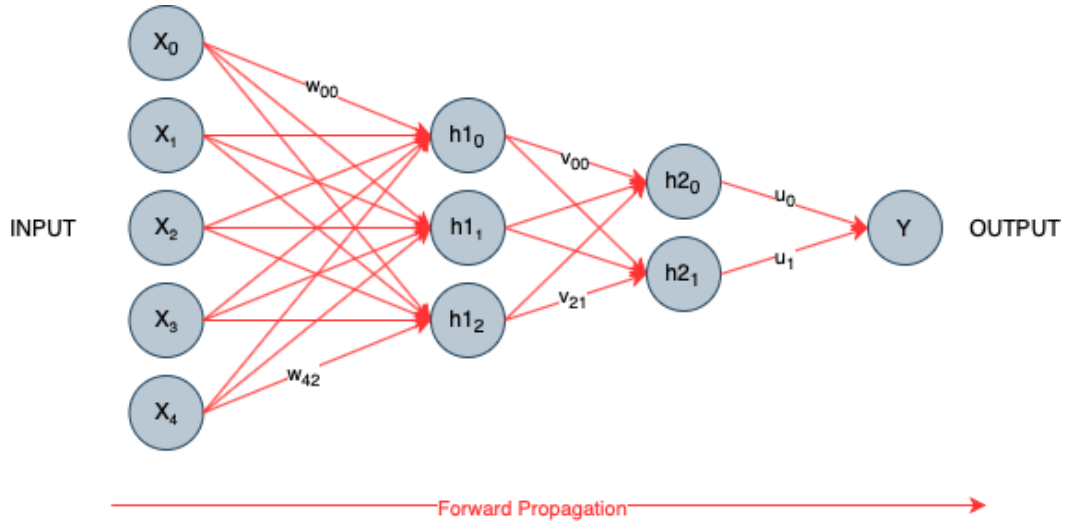

Figure 2.3: Flow of information during the forward propagation process in a deep neural network.

colored images that introduce a lot of diversity and complexity to the data and using this dataset benefits the system by generalizing on all sorts of information and data.

To understand the working of these various Deep Learning Models in terms of task of image denoising and enhancement, we need to understand the working of the forward and backward propagation stages. Figures 2.3 and 2.4 depict the flow of information during the forward and backward propagation stages of a vanilla deep neural network.

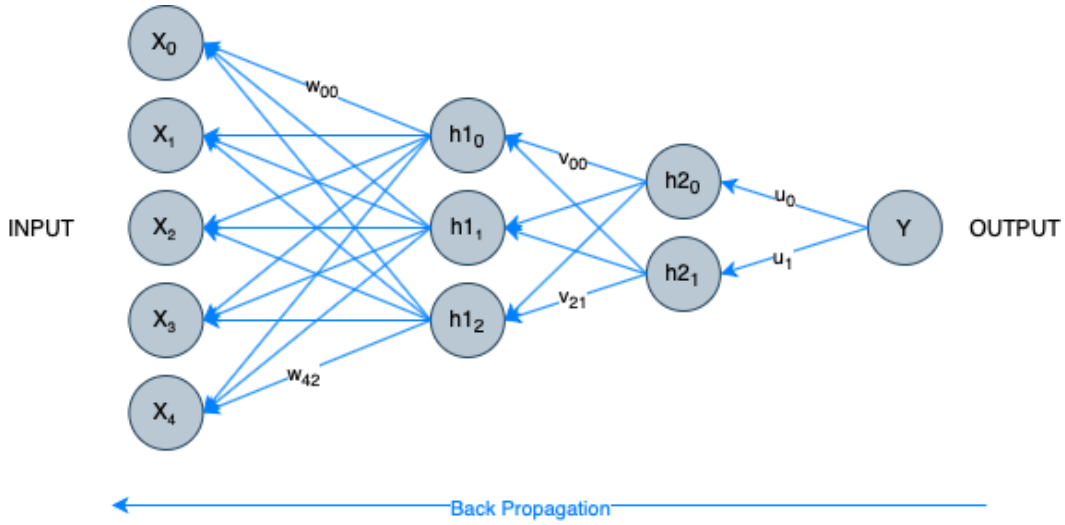

Figure 2.4: Flow of information during the backward propagation process in a deep neural network.

Equations 2.4, 2.5, 2.6, 2.7, 2.8, and 2.9 define the forward propagation process for each neuron in hidden and output layers as depicted in Figure 2.3.

$$h1_0 = A(w_{00} \cdot x_0 + w_{10} \cdot x_1 + w_{20} \cdot x_2 + w_{30} \cdot x_3 + w_{40} \cdot x_4) \tag{2.4}$$

$$h1_1 = A(w_{01} \cdot x_0 + w_{11} \cdot x_1 + w_{21} \cdot x_2 + w_{31} \cdot x_3 + w_{41} \cdot x_4) \tag{2.5}$$

$$h1_2 = A(w_{02} \cdot x_0 + w_{12} \cdot x_1 + w_{22} \cdot x_2 + w_{32} \cdot x_3 + w_{42} \cdot x_4) \tag{2.6}$$

$$h2_0 = A(v_{00} \cdot h1_0 + v_{10} \cdot h1_1 + v_{20} \cdot h1_2) \tag{2.7}$$

$$h2_1 = A(v_{01} \cdot h1_0 + v_{11} \cdot h1_1 + v_{21} \cdot h1_2) \tag{2.8}$$

$$Y = A(u_0 \cdot h2_0 + u_1 \cdot h2_1) \tag{2.9}$$

Where A is an Activation Function like Sigmoid, tanh, or ReLU.

Backward Propagation, as depicted in Figure 2.4, is expressed by Equation 2.10. This equation computes the gradient of the loss function with respect to the weights on the model's neurons.

$$w_{new} = w_{old} - \eta \cdot \frac{\partial L}{\partial w} \tag{2.10}$$

Where η represents the learning rate for the model, w represent the weights in the network, and L is the Loss function for the network.

Autoencoders have played a vital role in the efficient removal of noise from images [15]. Autoencoders, in general, is an unsupervised learning technique. It comprises an Encoder and a Decoder system that extract features and use those features to rebuild the image, respectively. In the Encoding stage, a bottleneck is imposed that forces the input data into a compressed knowledge representation of the input data. This compressed knowledge representation is then used to rebuild a cleaner version of the input data. A representation of an Autoencoder model is depicted in Figure 2.5.

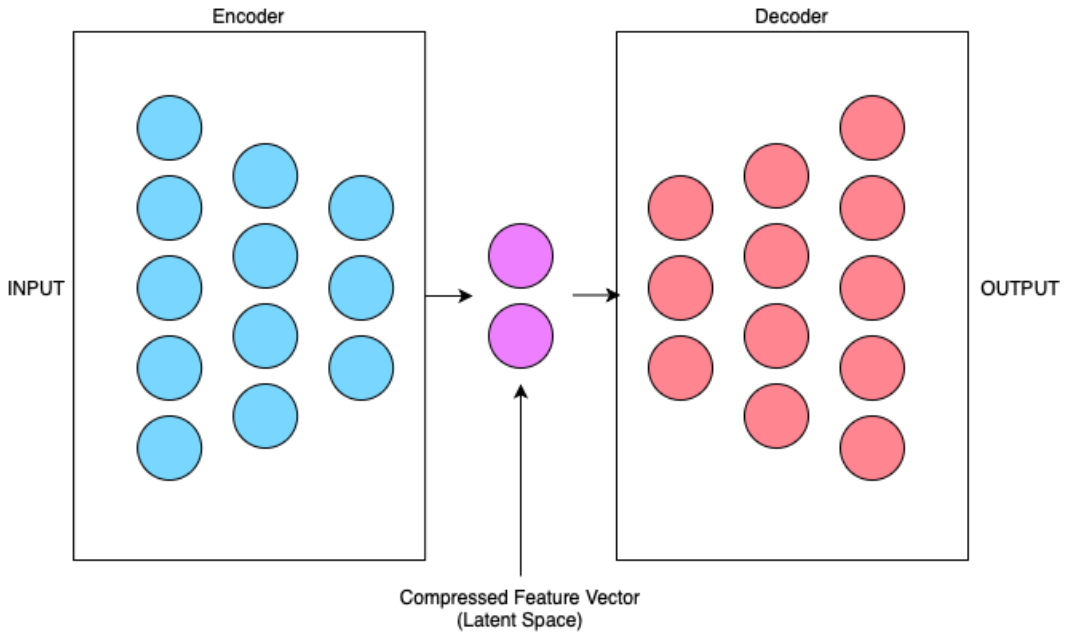

Figure 2.5: Generic Representation of an Autoencoder.

Deep Convolutional Neural Networks (CNNs) have also been very successful at the process of noise cleaning and enhancement in images [53]. A common practice followed here is that a system is trained to learn a specific type of noise by externally adding it to the image and letting it identify the structure of noise artifacts. Once these are learned, it becomes easier to remove those types of noise artifacts. Convolutional layers learn to extract relevant features that aid in optimal decision-making, such as Classification, as seen in Figure 2.6. To denoise noisy images, feature extraction is done to extract features that can be used to rebuild a cleaner version of the image.

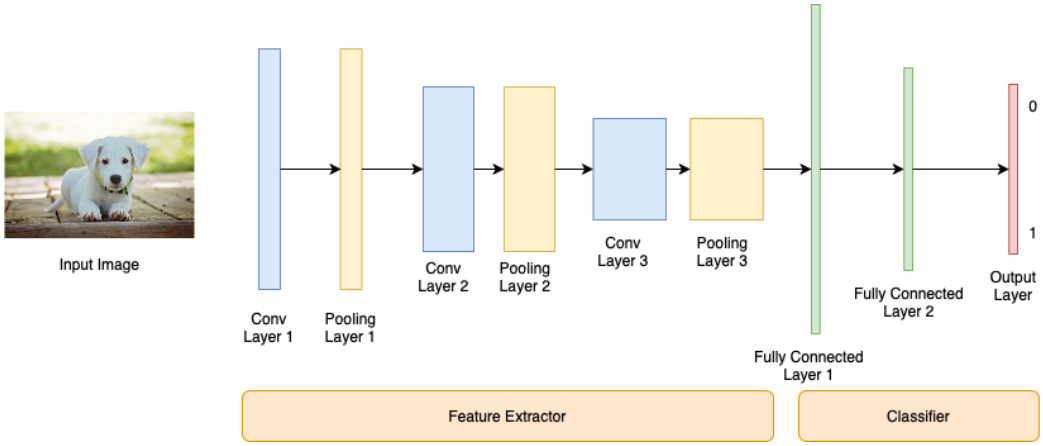

Figure 2.6: Generic Representation of a Convolutional Neural Network.

After Generative Adversarial Networks (GANs) were introduced in [17], they have been used to generate cleaner images, as seen in the paper presented by Yi et al. [58]. GANs make use of Game Theory to achieve their goal, wherein the Generator and Discriminator models contest each other as depicted in Figure 2.7. The Generator uses latent space mapping to generate a version of the image that the Discriminator model uses to predict if the image is real or not. For the task of image denoising, the Generator model tries to rebuild the image without noise, and the Discriminator model tries to distinguish it as a noisy or clean image, which in turn guides improvements in the denoising process.

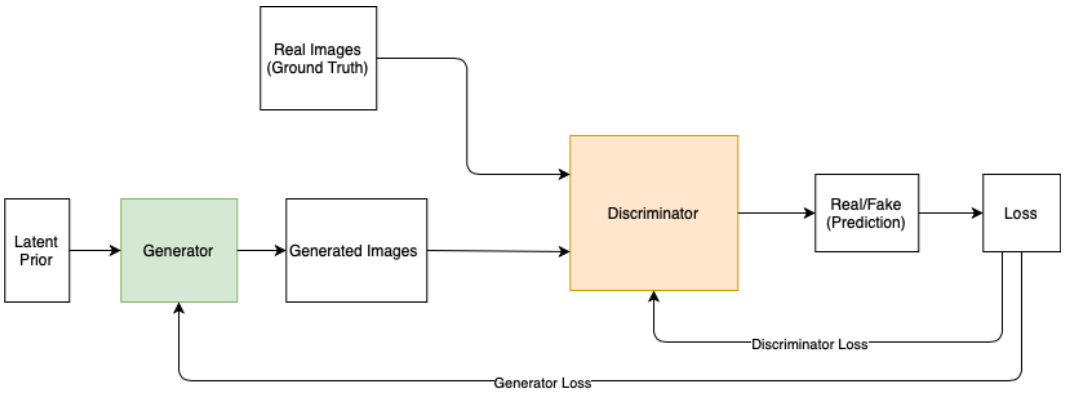

Figure 2.7: Generic Representation of a Generative Adversarial Network.

2.3 Segmentation Algorithms

Segmentation algorithms have, in the past, been used to identify objects and important context in images. Since understanding the context of an image is so crucial in image denoising if we could use a system to identify the essential areas of an image, it makes the process a lot easier. Ajmal et al. [1] talks about various segmentation-based algorithms that deal with instance segmentation and semantic segmentation. This paper also spoke about the Attention-based FCNs to denoise images that looked into some more depth by Haque et al. [19].

Ronnerberger et al. [46] introduced U-Nets in their paper. Their proposed architecture is easy to use on datasets that do not have segmentation masks associated with corresponding images. Once these segmentation maps are generated, these could be used to train the model further to tell noise from signal and vice versa. U-Nets are built with skip connections to avoid loss of any relevant and vital information, lost in the initial feature extraction

stages. These skip connections also prevent the vanishing gradients problem that affects models consisting of a deep structure.

2.4 Noise Models

Boyat et al. [3] talks about the various noise models that occur in digital images. Understanding the type of noise one is dealing with, along with the source of its occurrence, can help understand the underlying noise pattern. The Gaussian noise model is one of the models mentioned by them. This noise is distributed normally over a range of values. This type of noise could essentially cover various types of noise over the different ranges of distribution. Modeling this type of noise is essential as many different types of noise can be mapped to the pattern of Gaussian noise. Gaussian noise is additive noise that is represented as a probability density function equal to normal (Gaussian) distribution, as defined by Equation 2.11.

$$p(z) = \frac{1}{\sigma\sqrt{2\pi}} \cdot e^{\frac{-(z-\mu)^2}{2\sigma^2}} \tag{2.11}$$

Where,

- $p(z)$ is the Probability Density Function
- z represents the gray-level of the image
- μ represents the mean
- σ represents the standard deviation

- σ^2 represents the variance
- x represents

For the removal of multiplicative noise, such as speckle noise, found in ultrasound and MRI images, wavelet transformations have helped reduce it without affecting the sharpness or details in the image in the past [49]. Speckle noise typically gets introduced during the reconstruction phase of the imaging life cycle. Leal et al. [28] suggests using a non-linear wavelet transform to remove speckle noise from ultrasound images, which has worked successfully in recent times. As far as non-medical images are concerned, the occurrence is ubiquitous at varied intensities. The mere presence is enough to degrade an image and lose vital information from the image. Due to the presence of high-intensity components, low-pass filters have been an effective method of removing this type of noise. Modeling speckle noise to remove it from images is necessary as it can be representative of multiplicative noise.

Haque et al. [19] made use of the attention mechanism for the application of image denoising on the MNIST dataset [29]. Using the attention mechanism on smaller regions and multiple times would help learn and understand what bits of information are important for retaining context while discarding noise.

Traditionally, noise in images has been an aberration of visual data. With the introduction of new and more complex algorithms, comes this noise that minimally affects the image visually but extensively affects the structure

and context of the image. Fast Gradient Sign Attack (FGSM) [18] is an example of adversarial noise that harnesses the gradient of the image to trick the deep nets into identifying and classifying the image content. Equation 2.12 defines how this form of adversarial perturbations are added into an image. By understanding the pattern in which these perturbations occur and affect the context of an image, we are working towards building a blind denoising system that can handle and fix all occurrences of noise in images.

$$adv_x = x + \epsilon * sign(\nabla_x \cdot J(\theta, x, y)) \tag{2.12}$$

where,

- adv_x represents the Adversarial image
- x represents the original input image
- y represents the original input label
- ϵ represents the multiplier that ensures that the perturbations are small
- θ represents model parameters
- J represents the loss

2.5 Evaluation Techniques

To quantify the difference in change between the two images, before and after passing it through our proposed model, we need relevant metrics

that would score any improvement or deterioration in the image. We have used metrics that capture the structure of the image in both cases, while also using metrics that capture the effectiveness of a machine learning model by only changing the input to the model. An ideal evaluation technique would be where our proposed architecture could be exposed to various types of noise and image sizes. If any system works consistently in this setting, we would have an ideal and optimal blind denoising and enhancement system.

Making use of metrics like PSNR (Peak Signal-to-Noise Ratio) and SSIM (Structural Similarity Index Measure) are being used to evaluate changes in the structural changes of an image. Both these metrics use very different aspects of an image to determine the difference with respect to the original, non-noisy, ideal image. The lesser difference in the images equates to higher scores. PSNR primarily looks into the mean square error between the two images, whereas SSIM calculates changes in the luminance, contrast, and structure difference between them.

Metrics, like Precision, Recall, and F1-score, quantify any performance improvements in computer vision tasks like Object Detection and Classification. These metrics are scored using a confusion matrix that scores each class based on the number of times a class is correctly identified, correctly not identified, incorrectly identified, and incorrectly not identified. With metrics score more consistently than just using a metric that scores a system based on only how many classes are correctly or incorrectly identified.

Another metric that we considered to measure our system's effective-

ness was how the proposed model performed on unseen noise and adversarial noise. Building a system on one type of noise, added artificially to the image, may not be the best system to work with if it cannot generalize towards generic noise. Evaluating the effectiveness in a situation like this would justify the extensibility and scalability of the system being evaluated.

Most of the evaluation metrics score systems based on changes in visual features in the images. To quantify the effectiveness of any system, testing systems on adversarial noise is of utmost importance. The addition of adversarial noise, like Fast Gradient Sign Method (FGSM) noise, to an image, can confuse Convolutional Neural Networks (CNNs) [40] to misclassify images with high confidence. What is even more surprising is that there seems to be no visual change in the images. If a denoising and enhancement system can perform consistently on this type of noise, that is a positive performance in terms of an evaluation metric.

2.6 Hyperparameters

For training the machine learning models, two of the most common optimization algorithms I have come across in most of the papers are Stochastic Gradient Descent [47] and the Adam Optimizer [25]. These optimization algorithms help find parameter values required to minimize or maximize a loss function using the values of gradients in terms of parameters. Gradient Descent can be thought of as finding the value of a parameter that occurs at the minima or maxima, depending on the requirement.

Stochastic Gradient Descent, a version of the Gradient Descent algorithm, is used in cases where large datasets are involved. Here, the gradient descent procedure is run for each iteration, but an update to the coefficients of parameters is made for each training instance. With Stochastic Gradient Descent, there is a possibility of the gradient getting stuck in the local minima if the learning rate is very low. To overcome this problem, momentum is generally used with it. However, that would introduce an additional hyperparameter that would require tuning to achieve optimal results.

The Adam optimizer is an algorithm for gradient-based optimization of stochastic objective functions. It combines the advantages of two momentum-based SGD extensions, Root Mean Square Propagation (RMSProp) and Adaptive Gradient Algorithm (AdaGrad), and computes individual adaptive learning rates for different parameters and hence preferred over Stochastic Gradient Descent as suggested by in Kingma et al. [25].

2.7 Skip Connections

Skip Connections or Highway Networks are additional connections to one or more layers present past the immediate next layer of nodes. "Skip" or "highway" connections get the name from the functioning of this connection, as it always connects nodes by skipping or jumping at least one layer of nodes.

Srivastava et al. [48] introduced highway networks that help retain information and prevent the loss of information and prevent the diminishing gradient problem. He et al. [20] found that as the depth of the model in-

creases, the loss and accuracy start do not vary, but on training it further, the accuracy drops. They also talk about how overfitting was not the reason for the sudden drop in accuracy in [20]. In order to avoid this problem, He et al. propose a connection somewhat similar to a short circuit to pass information from the early layers to the later layers directly not to lose the relevance of information over the deep layers of the model. Orhan and Pitkow, in their paper [41], discuss multiple ways in which "skip connections eliminate singularities", where singularities are irregularities in the model performance.

Past research works, as seen in [33] and [54], have been published, which involve image denoising and enhancement, which successfully show how skip connections have benefited the architecture. Drozdzal et al. [12] discuss how these skip connections are important for their segmentation model for biomedical images. Tong et al. [54] and Mao et al. [33] have successfully proven that skip connections to their model leads to significant improvements in results by just introducing them to their respective models.

2.8 Loss Functions

Related researches [5, 11, 23, 34] have used either used loss functions usually like L1 Loss or Mean Square Error (MSE) Loss. For this thesis, we intend to use those Loss functions. We would be extending our work to Elastic Nets to study their effect on image denoising systems.

L1 Loss calculates the absolute difference between images, pixel-wise. This property is advantageous as this loss will try to minimize the pixel-wise

difference between the noisy image after denoising and the original image. In an end-to-end processing system, if I_{orig} is the original image and $I_{denoised}$ is the output of the model, then the loss would be defined by the Equation 2.13.

$$L_1 = \sum |I_{orig} - I_{denoised}| \tag{2.13}$$

L2 Loss or Mean Squared Error(MSE) computes the absolute squared difference between images. Unlike L1 loss, MSE works towards generating higher PSNR scores. This loss will minimize the pixel-wise difference between the noisy image after denoising and the original image, by penalizing incorrect pixels more heavily than those with values closer to the original image. In an end-to-end processing system, if I_{orig} is the original image and $I_{denoised}$ is the output of the model, then the loss would be defined by the Equation 2.14.

$$L_2(or MSE) = \frac{1}{n} \cdot \sum ||I_{orig} - I_{denoised}||^2 \tag{2.14}$$

In the paper by Zhao et al. [62], they have discussed the limitations of L2 loss. They discuss at length how it is not suitable in tasks involving image quality as L2 loss assumes that the noise in images is independent of local features. To utilize the benefits of both the loss functions, L1 loss and MSE, while offsetting their flaws, we can use Elastic Net loss, where ElasticNet loss is the barycentric weighted average of the L_1 and L_2 norms. It is defined by the Equation 2.15.

$$ElasticNet = \lambda \cdot L1 + (1 - \lambda) \cdot L_2 \tag{2.15}$$

Where λ needs to be tuned to get the best results and should lie between 0 and 1. If λ is 0, then the loss calculated will be L_2, and if λ is 1, the loss computed will be L_1.

Chapter 3

Methodology

To denoise and further enhance images, to understand the underlying image structure and context is of utmost importance. Segmentation algorithms have been used to provide the location of objects of importance in an image in the past. These algorithms are provided with enough information to learn the context of the image and identify which regions of the image were important for that task.

3.1 Hypothesis.

Our primary goal is to eliminate noise from images. To ensure only noise is eliminated, understanding the context of the image is required. The context of an image is extracted by passing the image through the U-Net [46] model, a Convolutional Neural Network whose primary task is to segment objects in an image. Here, we use this model to segment and identify noise artifacts in the image. The output of the U-Net model is used in two ways, with both methods being hypothesized and run independently.

In the first method, we made use of the output as an input to the outer model. In the second method, the output was attached to the noisy image as

an additional channel. Both the models only differ in this manner, while the remaining architecture remained consistent.

The segmentation model forms a part of an end-to-end system. For easier understanding, we will refer to the inner model as the U-Net model and the remaining model as the outer model. This outer model is a convolutional neural network. To ensure that the entire system trains optimally, without facing the vanishing or exploding gradients problem, we have introduced skip connections to this outer model. This helps ensure a robust working system.

To validate our hypothesis's correctness, we checked all changes in image quality by using the metrics, PSNR, and SSIM. We also used the Mixture of Gaussians to represent the presence of multiple sources of noise in images. Once we built our system using this artificial representation of noise, we tested the system on unseen noise such as Speckle noise, Gaussian noise, a mixture of Gaussian and Speckle noise, and Adversarial noise. With those experiments being run on prototypes trained on MNIST and Fashion-MNIST datasets, we were also able to extend it to ImageNet easily using transfer learning. Once we realized the potential for the scalability of our proposed system, we ran tests to evaluate the effectiveness of those scaled systems.

Apart from these tests, the processed "clean" images were be passed through various systems that performed various computer vision tasks like classification, object detection, and scene understanding to evaluate the effectiveness of the model. Evaluation of these images was done against the effectiveness of images that were originally passed through the systems, and

metrics for evaluation used here were metrics such as Precision, Recall, and F1-Score.

3.2 Approach

We created a prototype that would initially work on small datasets, which consisted of relatively smaller images. Once this prototype performed consistently through the evaluation experiments, we scaled the prototypes to work on a more complex dataset, ImageNet [8].

The original prototypes were also evaluated on the blind image denoising and enhancement task and evaluated their performance against visually unrecognizable adversarial noise.

The MNIST and Fashion-MNIST datasets were accessed primarily using PyTorch's [42] torchvision package [35], whereas the ImageNet's validation set was downloaded from one of their archived pages and used with the help of the torchvision package [35].

We used Python's open-sourced package - PyTorch [42] to build, train, and test our proposed architectures. Libraries like SciPy [24] and OpenCV [4] have functions to calculate the Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM) of an image, respectively. These functions were used to calculate each of the metric scores for every image. Our main point of comparison was between images before and after being passed through our proposed model. Any change in results was studied in-depth to understand

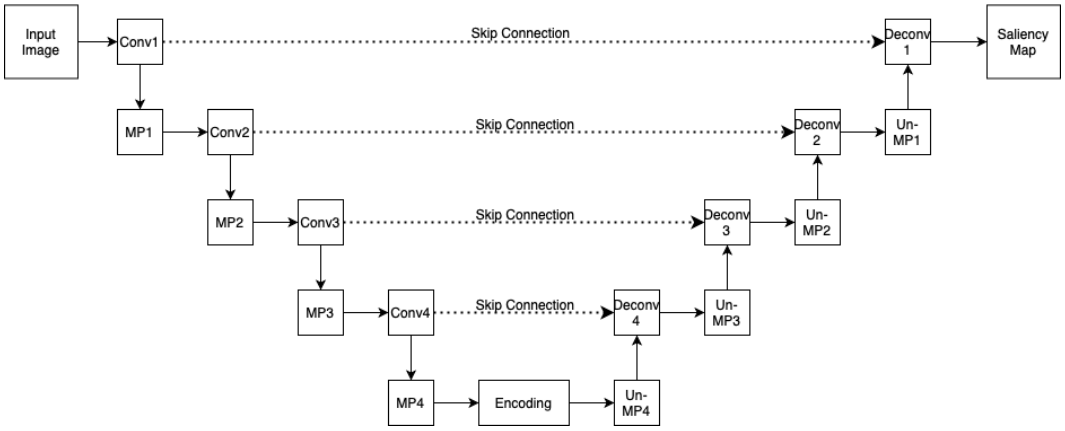

Figure 3.1: Inner U-Net Model [46].

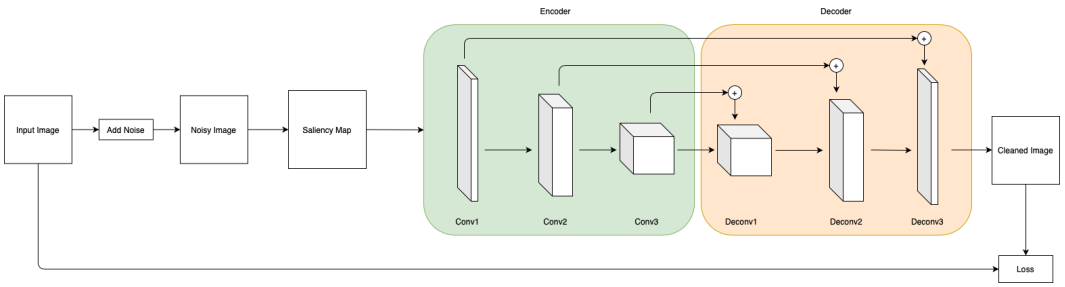

Figure 3.2: Baseline Encoder-Decoder Architecture without the inner segmentation U-Net model.

the reason for the change. We then made use of evaluation methods that helped us quantify the changes in images.

3.3 Architecture

The proposed architecture comprises of an inner model, U-Net, which is highlighted in blue and which is explained in detail later, along with an outer model. The outer model comprises of 2 regions - encoder(green) and

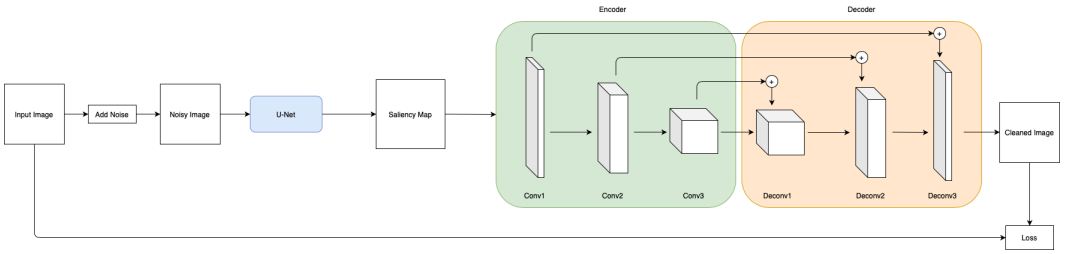

Figure 3.3: Proposed Architecture that uses the output of the inner segmentation U-Net algorithm, the Saliency Map, as input to the outer Encoder-Decoder Architecture.

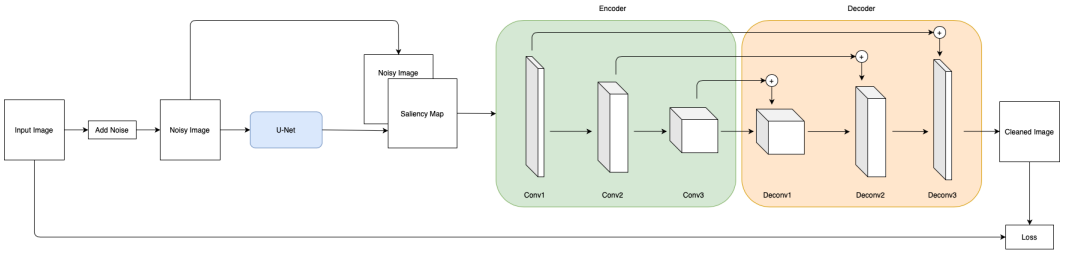

Figure 3.4: Proposed Architecture that uses the output of the inner segmentation U-Net algorithm, the Saliency Map, along with the Noisy Image added channel-wise as input to the outer Encoder-Decoder Architecture.

decoder(orange). Skip connections are added from the encoding layers to corresponding decoding layers as seen in Figures 3.2, 3.3, and 3.4, in order to prevent loss of information during the back-propagation stage. Figure 3.3 is one proposed version of architecture that uses the output of the inner U-Net model, the Saliency Map, as input to the outer Encoder-Decoder model. Figure 3.4 depicts another version of the proposed architecture that uses both the Saliency Map and the noisy image appended channel-wise as an input to the outer Encoder-Decoder model.

Figure 3.2 depicts a baseline version of the proposed architectures. This architecture lacks the inner U-Net model in comparison to the proposed architecture. We use this as a baseline to study the effect of the segmentation model, U-Nets, on the task of content-aware image denoising and enhancement.

The architecture that we proposed uses a segmentation architecture called U-Net [46]. Noisy images are passed as inputs to this model. The inputs get processed through convolutional layers "Conv" and Maxpooling layers "MP". This continues until the remaining information reaches the encoding layer. The up-sampling of information starts from there on, where information gets passed through the Deconvolution "Deconv" and Un-max-pooling layers "Un-MP". Skip connections are added at each level, where the output from a certain layer on the encoding side becomes an additional layer of input to the decoding side. These connections prevent loss of gradients as they relay more information and context to their adjoining layer. The output from this model

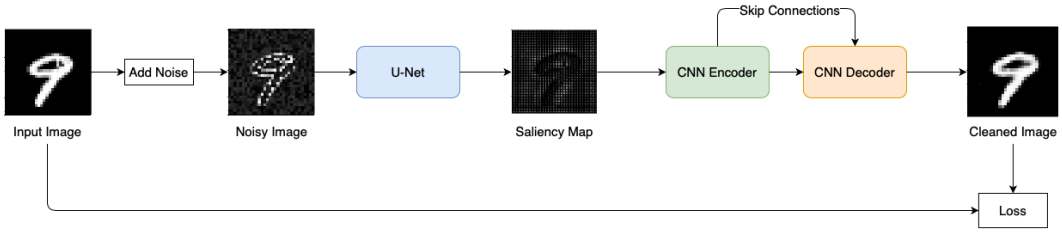

Figure 3.5: Example of image transformation at various stages of the proposed model, that uses the output of the U-Net model, the Saliency Map, as input to the outer Encoder-Decoder model.

is a 2-dimensional image, which retains the shape of the input image. This algorithm performs feature extraction in a larger end-to-end system, where primarily, features extracted are patterns of noise artifacts in terms of the content in the image.

A high-level description of the system we proposed comprises the inner segmentation model U-Net [46] connected to an external CNN-based system. This proposed model performs Context-Sensitive Image Denoising and Enhancement. The U-Net model forms a sub-model of a larger outer end-to-end image denoising system. For the outer model, we have chosen a Convolutional Neural Network-based encoding-decoding system with skip connections as seen in Figures 3.3 and 3.4. As far the inner segmentation model of U-Net goes, we have used the architecture as proposed by Ronneberger et al. [46] and is also depicted in Figure 3.1. A sample representation of what images looked like at various stages of both our proposed systems, is shown in Figures 3.5 and 3.6 respectively.

The use of the segmentation architecture U-Net [46] in the proposed

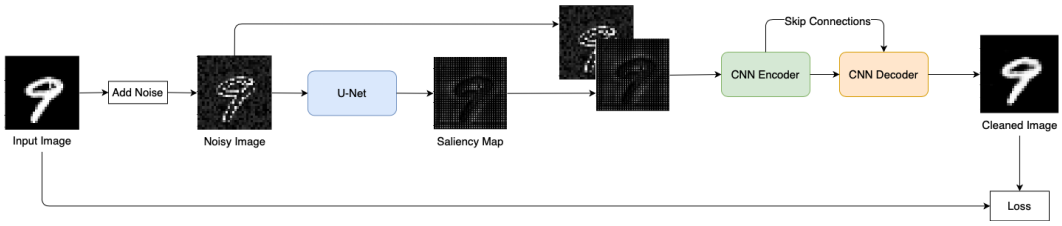

Figure 3.6: Example of image transformation at various stages of the proposed model, that uses the output of the U-Net model, the Saliency Map, along with the original noisy image appended channel-wise, as input to the outer Encoder-Decoder model.

system would allow the model to understand the context of the image, which would help maintain it. U-Nets are a preferred choice of segmentation architecture as they did not require masks to be provided during training as that internal model created its version of the mask, based on the task at hand. Once the model knows what is important and what is not, the process of denoising becomes easier. In essence, it tries to mimic the human visual system. The human visual system is very good at looking past the unimportant bits of information in an image and is trained to concentrate on the more important things. The human visual system understands the perspective of what is important and what is background noise.

We made edits to the U-Net architecture as defined by Ronneberger et al. [46] in terms of the addition and removal of layers and various activation functions but found the original architecture to perform best in comparison to all other variants. We have used 32, 64, 128, 256, 512 filters for each encoding layer to learn the various features from the input image to generate a Saliency Map representing noise patterns with respect to the image content.

Haque et al. [19] talks about using direct attention with a Convolutional Neural Network (CNN) [29] based encoder and LSTM [21] based decoder. Their analysis states that the attention mechanism works well for image denoising. In Ronneberger et al. [46], the U-Net model helps build the system to find the important parts of that image. We intend to use that aspect of the working of this model to detect the important bits of information in the image, and with that information, remove those noisy bits.

On including the output of the segmentation model U-Net [46], the outer model gets a better understanding of the structure of the image, which helps build more context as it allows the model to focus on the noise patterns in terms of the content in the image.

Chapter 4

Experiments

In this section, we look into the search space of various parameters that we have made use of in order to study the effectiveness of the proposed system. In each sub-section, we have discussed every parameter used and how each of them affects the effectiveness of the proposed system.

4.1 Types of Models

The architecture we are suggesting has two versions, which we intend on studying and analyzing. As a base model, we make use of an end-to-end autoencoder, which takes in a noisy image and returns a denoised and enhanced version. We use this model as our base as it forms the "outer" model of our architecture. We intend to make use of U-Nets to generate an image representing the noise artifacts in the image and the underlying image structure.

Figures 4.1 and 4.2, and 4.3 and 4.4 depict examples of the Saliency Map generated from the original image from the MNIST dataset [29], and Fashion-MNIST dataset [57] when noise was added to the original images respectively.

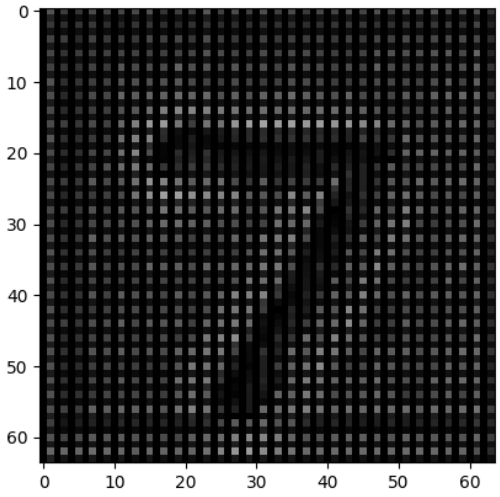

Figure 4.1: The Saliency Map of an image, depicting the number 7, taken from the MNIST dataset, after being processed through the U-Net segmentation model.

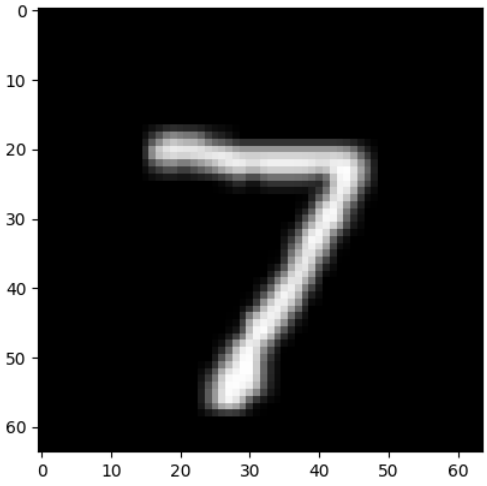

Figure 4.2: The Original image, depicting the number 7, taken from the MNIST dataset, before being processed through the proposed architectures.

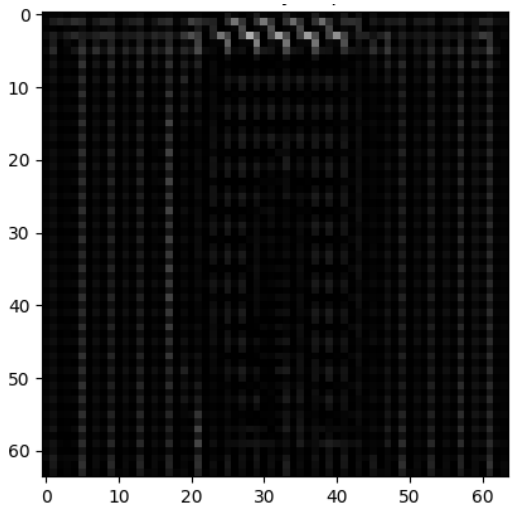

Figure 4.3: The Saliency Map of an image, depicting a pair of trousers, taken from the Fashion-MNIST dataset, after being processed through the U-Net segmentation model.

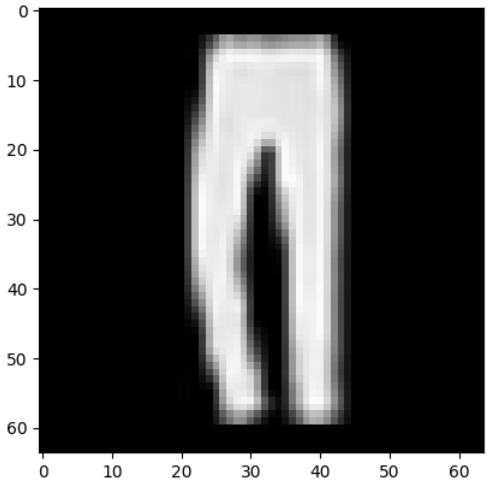

Figure 4.4: The Original image, depicting a pair of trousers, taken from the Fashion-MNIST dataset, before being processed through the proposed architectures.

The two versions of the proposed architecture come from utilizing the outputs of the segmentation model differently. The outputs of the segmentation model get used directly as an input to the outer autoencoder model, where the outer autoencoder model is trying to recreate a clean image from the noise artifacts and underlying image structure.

The other version of the proposed architecture uses the segmentation model's output and appends it to the noisy image as an additional channel and uses this image to recreate the clean image. Here, the goal is to let the model identify and map noise artifacts directly from the additional image channel and clean it in a context-sensitive manner using the underlying image structure.

The main difference between the two versions of the proposed architecture is that in the first version, the system relies on the Saliency Map generated from the input image, whereas the second version relies on the Saliency Map along with the original noisy image. The architectures for the two models are depicted in Figures 3.3 and 3.4.

4.2 Types of Datasets

For training and testing the denoising process of images, we use the MNIST [29] and Fashion-MNIST [57] datasets. The MNIST dataset was of paramount interest when created as they formed the basis for the training of Convolutional Neural Networks in the past. The Fashion-MNIST dataset was created to introduce more complexity into images of that size. Both of these datasets provide enough complexity to prototype our system. Finally, we introduced the ImageNet dataset to our prototype architectures to check if they scaled well to a complex dataset, consisting of thousands of everyday color images. The additional channels introduce more complexity than single-channel images while focusing on the object in question, just like MNIST and Fashion-MNIST. A sample of images from the various datasets are seen in Figures 4.5, 4.6, and 4.7 respectively.

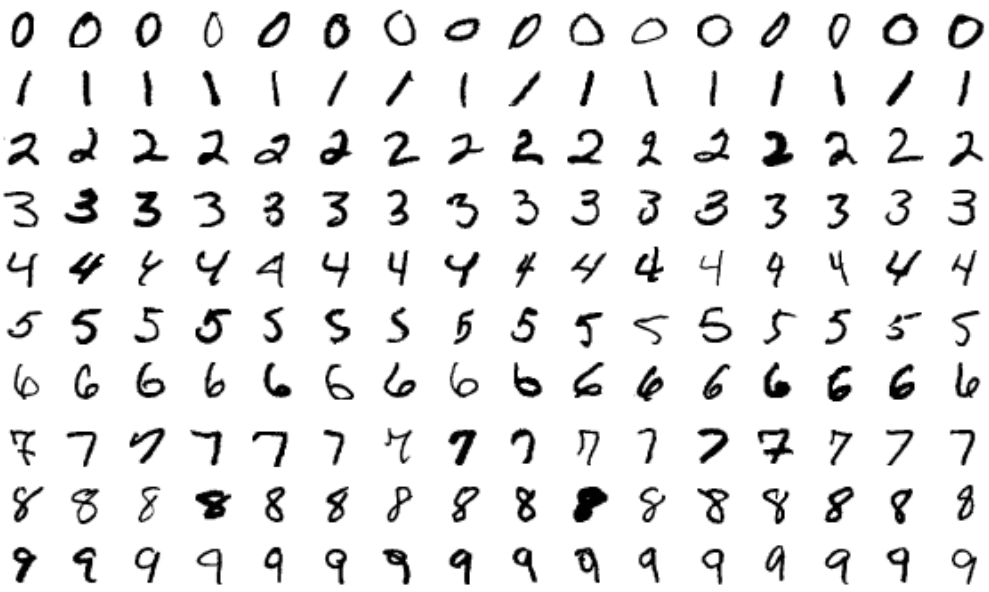

Figure 4.5: Examples of images taken from the MNIST dataset [29].

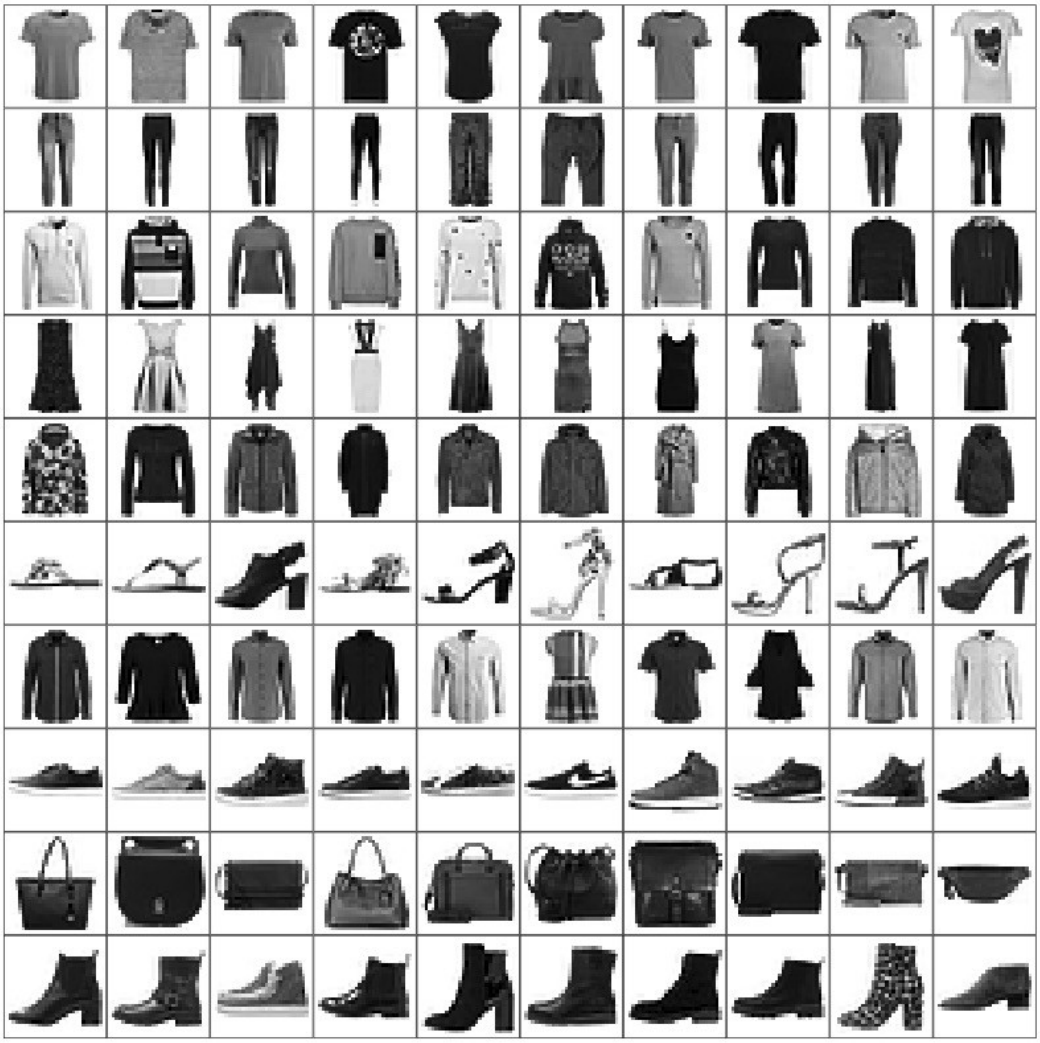

Figure 4.6: Examples of images taken from the Fashion-MNIST dataset [57].

Figure 4.7: Examples of images of dogs taken from the ImageNet dataset [8].

4.3 Image Sizes

For a thorough understanding of image structure, and to check for image size limitations, we train and evaluate our system on three image sizes

as mentioned below:

- 28 x 28 original image with padding of 2 on either side bringing the image size to 32 x 32.
- 56 x 56 original image with padding of 4 on either side bringing the image size to 64 x 64.
- 112 x 112 original image with padding of 8 on either side bringing the image size to 128 x 128.

As mentioned earlier, we worked with the image sizes to check the effectiveness of the proposed system on MNIST [29], when the images were resized from 32x32 to 64x64 and 128x128, respectively. By doing this, we were trying to study how noise, generated by image resizing, affects our system's effectiveness. MNIST and Fashion-MNIST both were resized to the image sizes mentioned above to check if the noise generated by the resizing of a more complex dataset affects the system more than noise generated from resizing a relatively more uncomplicated dataset.

4.4 Noise Types

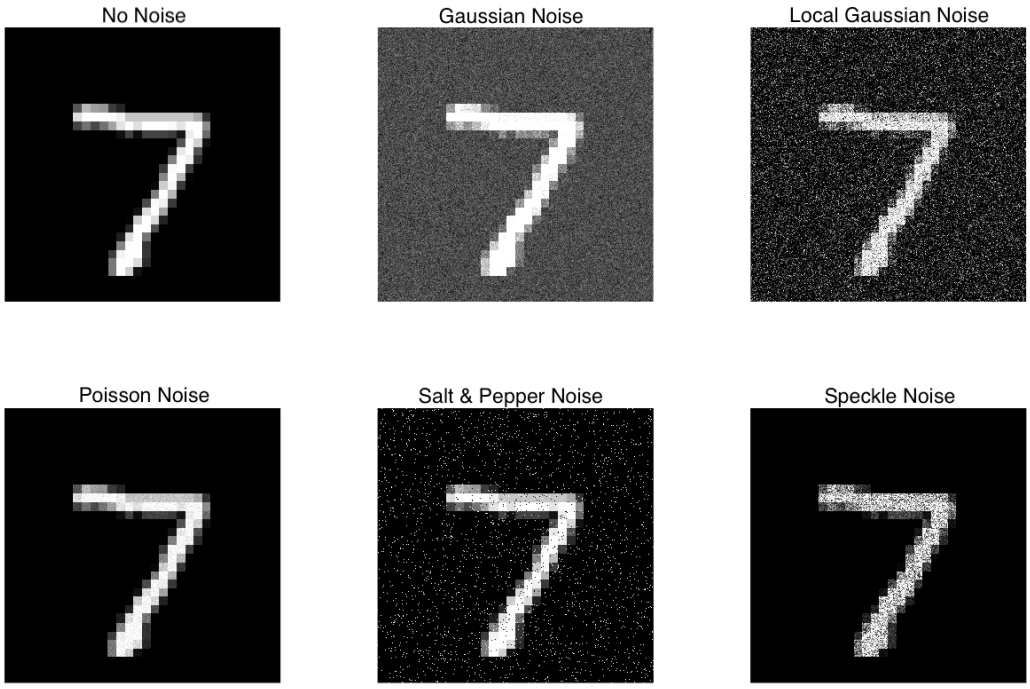

Figure 4.8: Effect of Gaussian, Local Gaussian, Poisson, Salt and Pepper, and Speckle Noise, when added to an image. This is a horizontal representation of Figure 1.1.

Creating a universal image denoising system is a hard problem to solve. Understanding the various patterns formed by the different noise types gave an insight into solving this problem differently. Figure 4.8 shows how various types of noise affect images.

On closely observing Figure 4.8, we can notice the following:

1. Gaussian Noise and Salt and Pepper Noise can be represented as a version of Local Gaussian Noise.

2. Poisson Noise can be represented as a version Speckle Noise.

3. Speckle Noise can be represented as a function of Gaussian Noise.

On observing how a Gaussian can represent most types of noise, we tried to model noise as a Mixture of Gaussians (MoG). These Gaussians were empirically defined, such that they represented a combination of the various noise types, as seen in Figure 4.8. We made use of two Gaussian distributions, one that was high in intensity but with a low standard deviation and the other one had lower-intensity but higher standard deviation, to represent the various types of noise types.

Choosing types of noise which the proposed system learns to understand is crucial in deciphering noise universally. We have made use of Gaussian noise, speckle noise, a combination of Gaussian and speckle noise, noise represented as a mixture of Gaussians, and adversarial noise to study the trained model's effectiveness on unseen noisy images. On getting a system to perform optimally on these various types of noise, we build a system that could be scaled to blind content-aware denoising and enhancing system. Figure 4.9 depicts multiple types of noise to an image of number 7, taken from the MNIST dataset [29]. Gaussian noise, Speckle noise, a combination of Gaussian and Speckle noise, noise modeled as a Mixture of Gaussians, and Adversarial (FGSM) noise were added to the original image are depicted in Figure 4.9.

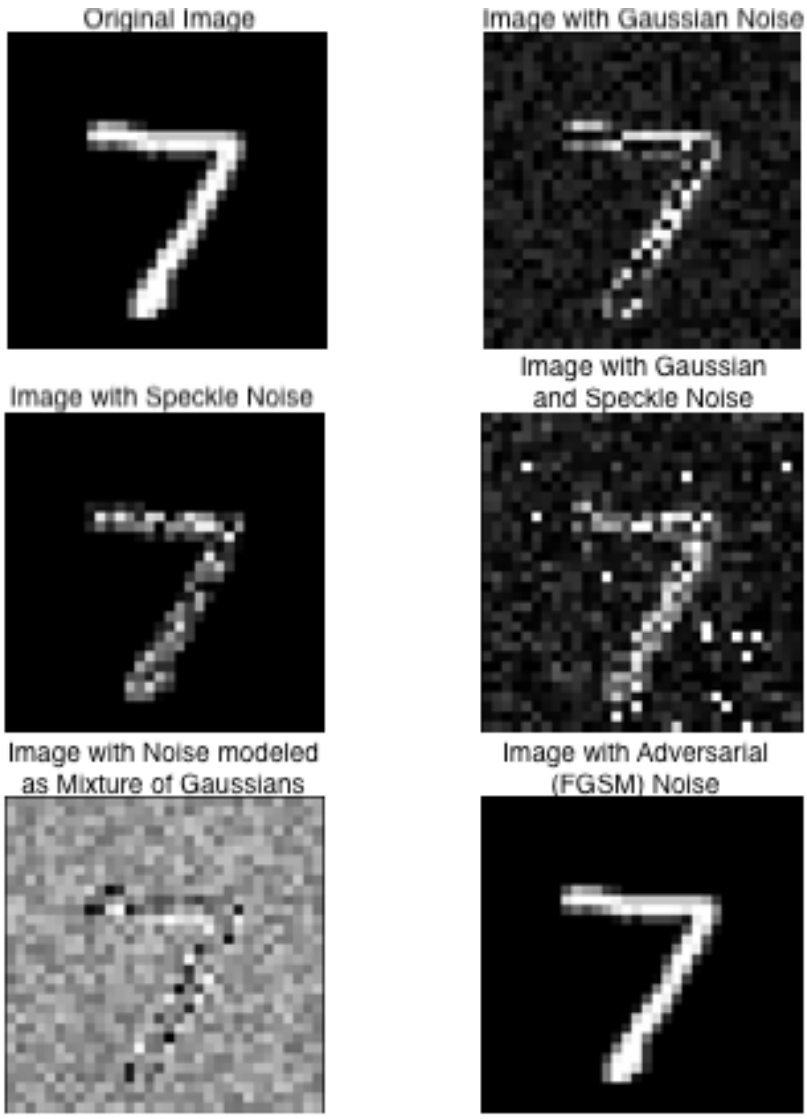

Figure 4.9: Effect of different type of noise on the original image of a number 7, taken from the MNIST dataset. Gaussian noise, Speckle noise, a combination of Gaussian and Speckle noise, noise modeled as a Mixture of Gaussians and Adversarial (FGSM) noise added to the original image.

4.5 Types of Losses

To train our systems, we have made use of various types of loss functions to test which loss types help speed up the convergence process while retaining its performance accuracy. We have made use of the L1 loss, L2 or MSE loss, and ElasticNet loss with a λ of 0.25, 0.5 and 0.75 as seen in the Equations 4.1, 4.2, and 4.3

- L1 Loss:

$$L_1 = \sum |I_{orig} - I_{denoised}| \tag{4.1}$$

  where I_{orig} is the original image and $I_{denoised}$ is the output of the model

- Mean Squared Error (MSE) or L_2 Loss:

$$L_2(or MSE) = \frac{1}{n} \cdot \sum ||I_{orig} - I_{denoised}||^2 \tag{4.2}$$

  where I_{orig} is the original image and $I_{denoised}$ is the output of the model

- Elastic Loss with $\lambda = 0.25$, 0.5, and 0.75:

$$ElasticNet = \lambda \cdot L1 + (1 - \lambda) \cdot L_2 \tag{4.3}$$

4.6 Optimizer Types

We have made use of two types of optimizers in our training and testing process - Stochastic Gradient Descent and Adam Optimizer. We found that both optimizers perform optimally, but Adam optimizer was a better optimizer

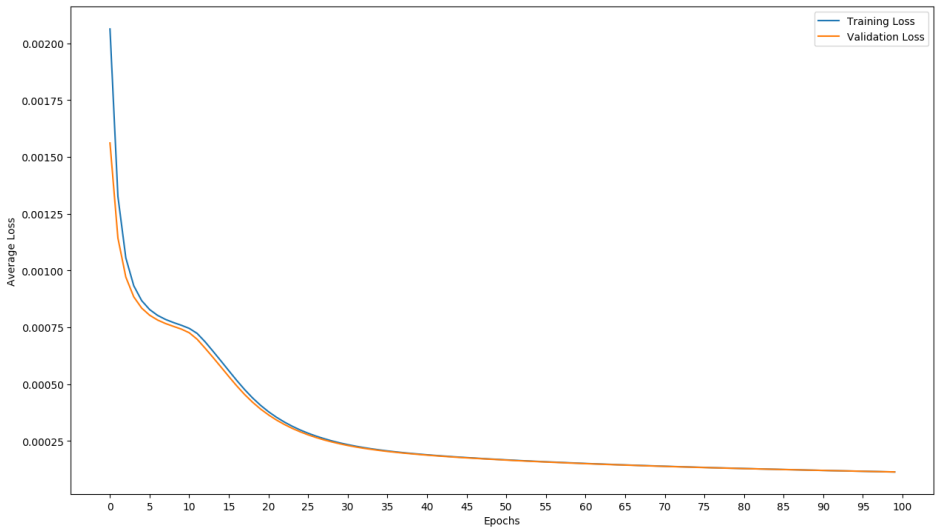

Figure 4.10: Loss curve of the Baseline Encoder-Decoder Model when it is trained with the Adam optimizer.

to use as the model converged faster. The hyperparameter tuning was only limited to the learning rate, unlike in the case of Stochastic Gradient Descent, which required tuning for momentum along with learning rate, as seen in Figures 4.10 and 4.11 respectively.

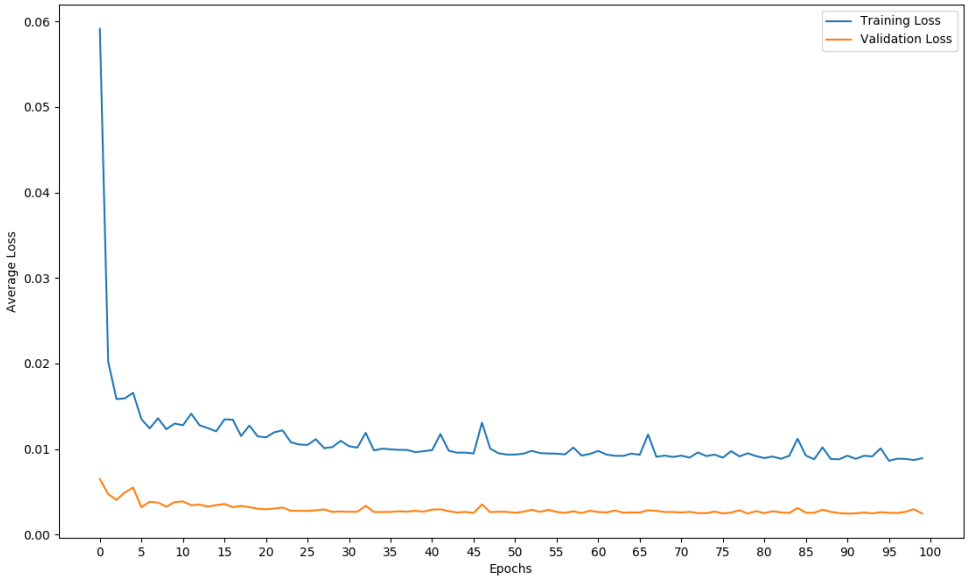

Figure 4.11: Loss curve of the Baseline Encoder-Decoder Model when it is trained with the Stochastic Gradient Descent optimizer.

4.7 Skip Connections

For the outer end-to-end autoencoder model, we propose the addition of skip connections as one version and the absence as another.

Skip connections, also referred to as highway networks, connect from one layer to another, but only by skipping at least one layer. This connection helps retain gradients during the backpropagation step, and they are extremely beneficial in cases where the layers are deep, where the chance of vanishing gradients is possible. This lets us understand if the facilitation of gradients with skip connections helps speed up convergence and if it affects the working performance of the proposed model.

In Figure 4.12 we notice how quickly the gradients converge in comparison to Figure 4.13 where the gradients bounce back and forth, causing the loss to vary, instead of converging.

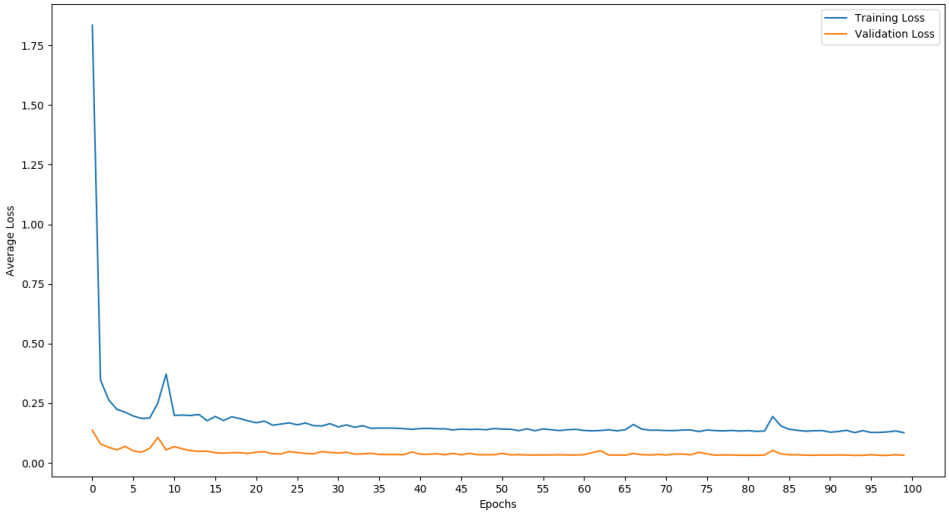

Figure 4.12: Loss curve of the Baseline Encoder-Decoder Model when skip connections were added to the architecture.

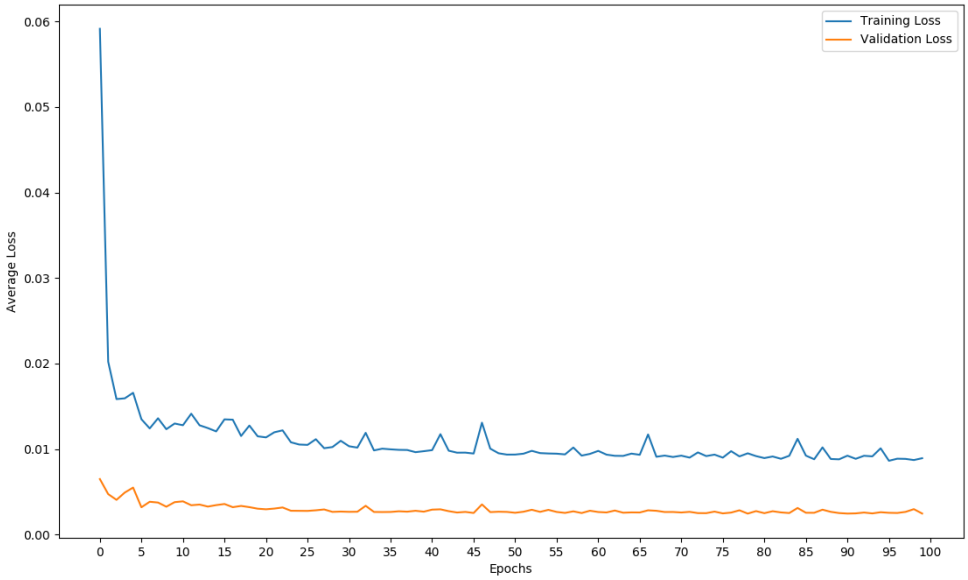

Figure 4.13: Loss curve of the Baseline Encoder-Decoder Model when skip connections were not a part of the architecture.

Chapter 5

Results and Discussion

5.1 Evaluation Process

For the evaluation process of the images, we made use of the following:

1. Evaluating the change in image quality with the help of metrics such as PSNR and SSIM.

2. Evaluating the effectiveness of computer vision tasks like object detection and classification when the images are denoised and enhanced using our proposed architectures.

3. Evaluating the effectiveness of the proposed system on unseen noise, where the noise was modeled as a Mixture of Gaussians (MoG).

4. Evaluating the effectiveness of the proposed system, trained on noise modeled as a Mixture of Gaussians, on visually invisible Adversarial noise (FGSM).

5. Evaluating the effectiveness of the proposed architectures, trained on MNIST [29] and Fashion-MNIST [57] datasets with noise modeled as a Mixture of Gaussians.

The above-mentioned evaluation processes highlight different aspects to assess the proposed system. We look into each of the evaluating processes in depth below:

1. Use of signal-structure metrics to evaluate changes in the image quality. The metrics used were:

   (a) **PSNR (Peak Signal-to-Noise Ratio)** : This term is defined as a ratio between the maximum possible signal strength and the amount of noise that affects the quality of an image. This ratio is calculated in the logarithmic scale due to existence of wide dynamic range in images. The formula to calculate PSNR is expressed below:

$$PSNR = 20 \cdot log_{10} \frac{SIG_{max}}{\sqrt{MSE}} \tag{5.1}$$

$$MSE = \frac{1}{mn} \cdot \sum_{rows=1}^{m} \sum_{columns=1}^{n} ||I_{orig} - I_{degraded}||^2 \tag{5.2}$$

   where,

   - SIG_{max} is the maximum signal strength of the original image
   - MSE is the Mean Squared Error
   - I_{orig} is the image data of the original image
   - $I_{degraded}$ is the image data of the degraded image in question
   - m is the number of rows in the image
   - n is the number of columns in the image

(b) **SSIM (Structural Similarity Index)** [56]: It is a method to measure the similarity between two images in terms of factors like contrast, luminance, and structural context. It is viewed as a quality measure for an image being compared to the original image, which in these cases, is regarded as an ideal image. This metric evaluates the images on structural metrics instead of the absolute difference in pixel values, as seen in PSNR. According to Wang et al. [56], SSIM can be calculated using the following formulae:

$$SSIM(x, y) = [l(x, y)]^{\alpha} \cdot [c(x, y)]^{\beta} \cdot [s(x, y)]^{\gamma} \tag{5.3}$$

$$l(x, y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1} \tag{5.4}$$

$$c(x, y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2} \tag{5.5}$$

$$s(x, y) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3} \tag{5.6}$$

where,

- x is the degraded image in question
- y is the original image
- l is the luminance term
- c is the contrast term

- s is the structural term
- μ_x is the local means of the degraded image
- μ_y is the local means of the original image
- σ_x is the local standard deviation of the degraded image
- σ_y is the local standard deviation of the original image
- σ_{xy} is the local cross-covariance for the images x and y

if $\alpha = \beta = \gamma = 1$ and $C_3 = C_2/2$, we get the following formula:

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{5.7}$$

2. We made use of pre-trained models, fitted to perform optimally on the dataset in question in the following manner:

   (a) **Classification**: the process of reading an image, analyzing it to map the various objects in the image to a certain class depending on patterns found on that object. We would be making use of the model as defined by Byerly et al. in their paper [6].

   For this process, we used a pre-trained ResNet-based Convolutional Neural Network (CNN) fitted to perform optimally on the Fashion-MNIST dataset. We then passed the input images through our proposed network and then through the classifier, as mentioned above, and recorded metrics like Precision, Recall, and F1-scores in both cases.

i. **Precision** : It represents the fraction of correctly predicted outputs to the total predicted positive outputs. It is computed using the following formula:

$$Precision = \frac{Number\ of\ True\ Positives}{Number\ of\ True\ Positives + Number\ of\ False\ Positives} \tag{5.8}$$

ii. **Recall** : It represents the fraction of correctly predicted outputs to all the observations in the actual class. It is computed using the following formula:

$$Recall = \frac{Number\ of\ True\ Positives}{Number\ of\ True\ Positives + Number\ of\ False\ Negatives} \tag{5.9}$$

iii. **F1 score** : This score is a measure of the evaluation tests' accuracy. It is the weighted harmonic mean of the above defined Precision and Recall as shown below:

$$F1\ Score = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{5.10}$$

(b) **Object detection**: the task of detecting instances of objects of a certain class within an image.

For this process, we used an R-CNN-based, and InceptionNet [50] based pre-trained network fitted to perform optimally on the dataset

in use, and then record IoUs and mAP scores in both cases to compare the effectiveness of the change in inputs brought about by our proposed network.

(c) **Scene understanding**: this computer vision is a challenging task, but has critical applications in autonomous driving and virtual reality as seen in [59]. The architecture discussed in this paper is what we will be using to test this application.

3. By studying the effect of the proposed model on unseen noise, we made tested the extent to which our system performed blind denoising and enhancement. We built this system by replicating the occurrence of multiple patterns of noise captured during the image processing pipeline and caused by the environment. This was done with the help of a mixture of Gaussians and tested on multiple types of noise unseen by the model during training - Speckle noise, Gaussian noise, a Mixture of Gaussians, and Speckle noise, and adversarial noise.

4. By evaluating the effectiveness of the proposed system on Adversarial noise, like FGSM noise, delves into how well the proposed system performs in terms of context-sensitive image denoising and enhancement. Since adversarial noise is invisible to the human eye, the removal of this noise helps prevent state-of-the-art models to not suffer in effectiveness, affected by noisy inputs.

5. Studying and evaluating the effect of scaling the prototype to a larger

and more complex dataset like ImageNet [8] that has more multiple channels instead of single-channel images as seen in MNIST [29] and Fashion-MNIST [57].

Stopping criteria for the proposed architecture could be when improvements in image quality would provide better results on the same model. By understanding how the image structure and context change after passing them through our proposed network would also give insight into where the system would fail to work as expected.

5.2 Results

The evaluation plans mentioned above were tried, and the results for each of those tests are as seen below. Tables 5.1 and 5.2 highlights the results of metrics, PSNR and SSIM, on the MNIST and Fashion-MNIST datasets when the outer model had no skip connections and had skip connections respectively. In Tables 5.1 and 5.2, "AE", "SAL" and "CONCAT" reference the baseline autoencoder, proposed architecture using the output of the inner segmentation model (Saliency Maps), and the proposed architecture using the output of the inner segmentation model (Saliency Maps) along with the original noisy image respectively.

With the absence of skip connections in the models, trained on MNIST, PSNR scores increased from 25.6080, for the baseline architecture, to 27.5111 and 28.1987 whereas SSIM scores increased from 0.9529, for the baseline ar-

chitecture, to 0.9611 and 0.9672 for the two proposed systems respectively as seen in Table 5.1. When testing the system, trained on Fashion-MNIST, PSNR scores increased from 19.5992 to 19.7682 and 21.6835, whereas SSIM scores increased from 0.6700 to 0.6902 and 0.7986 for the two proposed architectures. The absence of skip connections in the models led to the extraction of sub-optimal features that, in turn, led to blank reconstructions of the images, as seen in Figures 5.1 and 5.2.

Similarly, when skip connections were added as a part of the baseline and proposed architectures, testing models trained on MNIST increased PSNR scores from 31.0362 to 35.1684 and 36.0127, and SSIM scores increased from 0.9632 to 0.9927 and 0.9930, where the highest scores correspond to the proposed architecture that uses the output of the inner segmentation U-Net model along with the original noisy image as input for the reconstruction process of the denoised image. On testing the models trained with the Fashion-MNIST dataset, PSNR scores increased from 24.2192 to 27.9600 and 31.2166, and SSIM scores increased from 0.9116 to 0.9351 and 0.9530. Regardless of the addition of skip connections to the architecture, the PSNR and SSIM scores are higher than those of the baseline architecture, where the model using the Saliency Map along with the noisy image as input to the outer Encoder-Decoder system performs more effectively than its counterpart. These results are observed in Table 5.2. Overall, architectures built with skip connections on MNIST, and Fashion-MNIST dataset had the best PSNR scores of 36.0127 and 31.2166, and the best SSIM scores of 0.9930 and 0.9530, respectively.

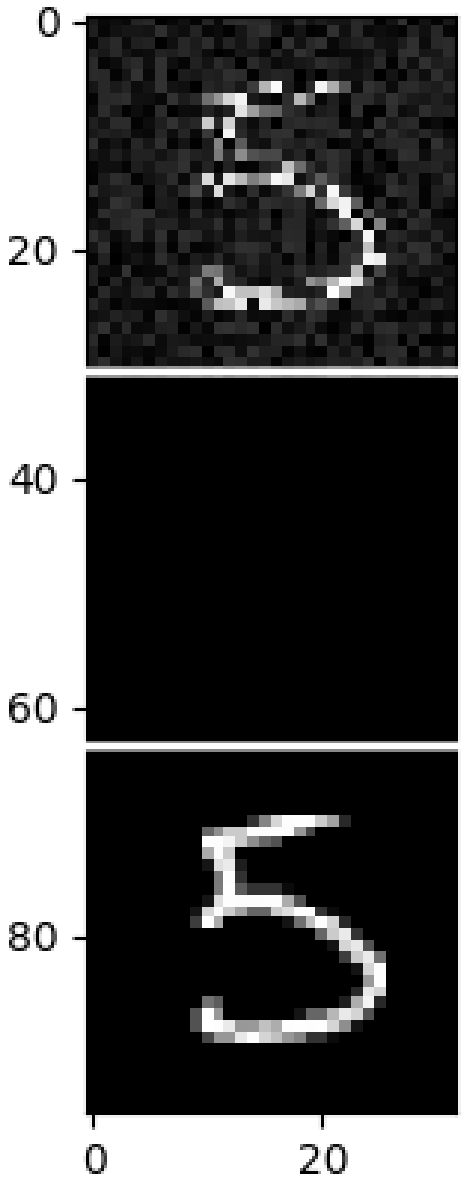

Figure 5.1: Instances where the proposed model reconstructed a blank image, when training on the MNIST dataset. Here, the top image is the noisy image, middle image is the expected denoised image as an output from the proposed model, and the bottom image is the original image.

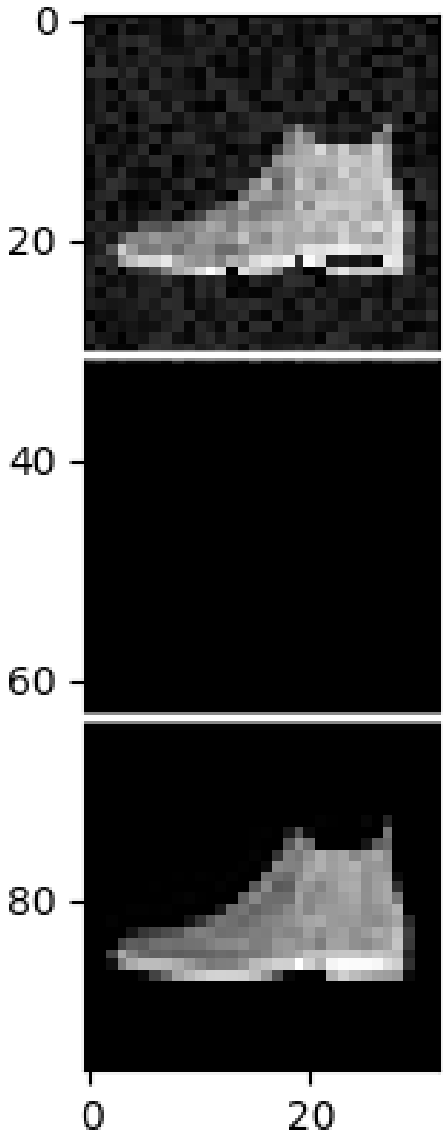

Figure 5.2: Instances where the proposed model reconstructed a blank image, when training on the Fashion-MNIST dataset. Here, the top image is the noisy image, middle image is the expected denoised image as an output from the model, and the bottom image is the original image.

Table 5.1: Performance of the Baseline Architectures and the two proposed Architectures, when Skip Connections were not a part of the Outer Encoder-Decoder Architecture.

Dataset	Metric	Image Type	AE	SAL	CONCAT
MNIST	**PSNR**	Noisy Images	18.7520	18.7511	18.7536
		Denoised Images	25.6080	27.5111	28.1987
	SSIM	Noisy Images	0.2990	0.2990	0.2990
		Denoised Images	0.9529	0.9611	0.9672
Fashion-MNIST	**PSNR**	Noisy Images	18.7522	18.7493	18.7537
		Denoised Images	19.5992	19.7682	21.6835
	SSIM	Noisy Images	0.4353	0.4354	0.4355
		Denoised Images	0.6700	0.6902	0.7986

Table 5.2: Performance of the Baseline Architectures and the two proposed Architectures, when Skip Connections were added as a part of the Outer Encoder-Decoder Architecture.

Dataset	Metric	Image Type	AE	SAL	CONCAT
MNIST	**PSNR**	Noisy Images	18.7509	18.7519	18.7517
		Denoised Images	31.0362	35.1684	36.0127
	SSIM	Noisy Images	0.2990	0.2990	0.2990
		Denoised Images	0.9632	0.9927	0.9930
Fashion-MNIST	**PSNR**	Noisy Images	18.7523	18.7540	18.7532
		Denoised Images	24.2192	27.9600	31.2166
	SSIM	Noisy Images	0.4354	0.4354	0.4354
		Denoised Images	0.9116	0.9351	0.9530

On delving deeper into the results of the proposed architectures, the model that takes the U-Net model's output, Saliency Maps, and the original noisy image performs better than the proposed architecture that only uses the Saliency Maps for the reconstruction of denoising image. We believe that the introduction of the original noisy image in the reconstruction process supplements the process by introducing more context of the image. The U-Net

segmentation model identifies noisy pixels in terms of the content of the image. When this additional information gets added channel-wise, it aids the model in building a system to correctly identify and remove noise while retaining the context of the image.

While running our experiments, we made use of various loss functions - L_1 loss, L_2 loss and ElasticNet loss using the L_1 and L_2 loss with λ of 0.25, 0.5 and 0.75. In the majority of the cases, models using the L_2 loss function perform consistently and optimally. When any of the other loss functions were used, they were either inconsistent with their results or generated blank reconstructions of the image. Introducing skip connections into the proposed architecture help reduce such instances.

We initially trained our baseline architectures with Stochastic Gradient Descent(SGD) as an optimizer, which took in learning rates and momentum values. Once this model was tuned to perform optimally, we did not change any hyperparameters to analyze the effectiveness of different architectures under the same conditions. On running all experiments with Stochastic Gradient Descent(SGD), we ran all the experiments using Adam optimizer. The Adam optimizer combines the advantages of two other momentum-based Stochastic Gradient Descent(SGD) extensions, Root Mean Square Propagation (RMSProp), and Adaptive Gradient Algorithm (AdaGrad), where it computes individual adaptive learning rates for different parameters. Hence, we only made use of Adam optimizer to run our tests, instead of using the RMSProp and AdaGrad optimizers.

With the introduction of Adam optimizer, the models converged faster than they did with Stochastic Gradient Descent(SGD) while maintaining the models' effectiveness. Models trained with the Stochastic Gradient Descent (SGD) optimizer did not perform consistently, whereas models trained with the Adam optimizer had performed consistently. Instances where the models did not optimally converge produced outputs, as seen in Figures 5.1 and 5.2, where a model trained with Stochastic Gradient Descent (SGD) failed to generate any output.

While running our various experiments, we noticed that the models converged well before the 100^{th} epoch for the MNIST and Fashion-MNIST datasets. For that reason, we introduced an early stopping criterion for all our models. If the validation loss did not significantly reduce over a fixed set of epochs (15, arbitrarily picked), the program would save parameter values that produce the lowest loss value, and the model would stop training. If the early-stopping mechanism was not triggered, the model would end after training at the end of 100 epochs.

We also ran experiments by changing the image size of the MNIST and Fashion-MNIST datasets from 32x32 (with padding) to 64x64 and 128x128. Images were upsampled using the Bilinear Interpolation mechanism, instead of the Bicubic interpolation as the speed of execution became a vital factor to consider, with a very little difference in effectiveness. The models' effectiveness remained consistent even when the size of the images was changed, proving that the proposed algorithm would work efficiently over various image sizes.

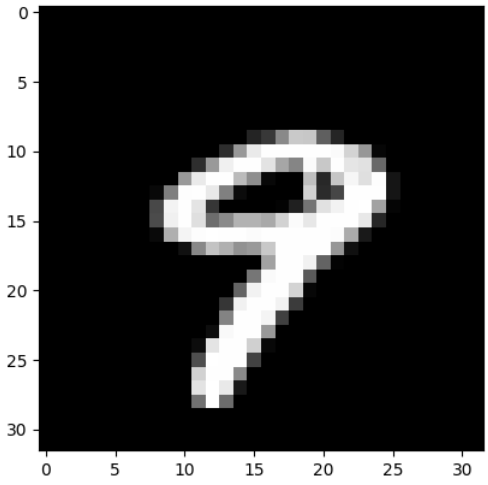

Figure 5.3: Instance of an Image from the MNIST [29] dataset, depicting number 9.

The proposed architecture, which used the original noisy image with the Saliency Map, generated by the inner U-Net model, generally performed better than the other proposed model, and both proposed models performed better than the baseline models.

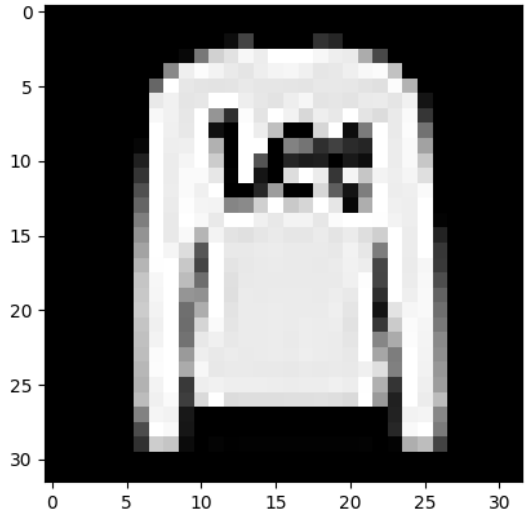

Figure 5.4: Instance of an Image from the Fashion-MNIST [57] dataset, depicting a pullover.

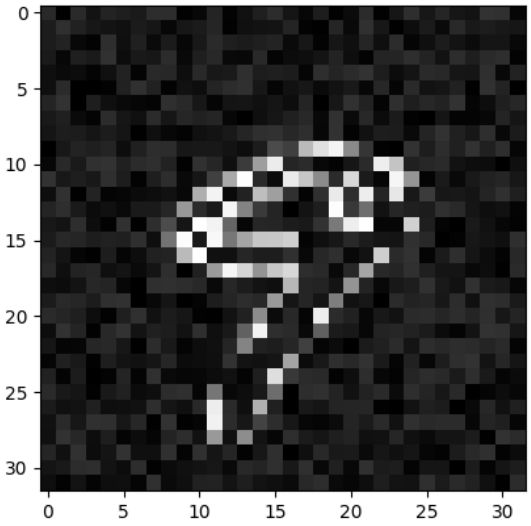

Figure 5.5: Instance of an Image from the MNIST [29] dataset, depicting number 9, as seen in Figure 5.3, with noise added to the image.

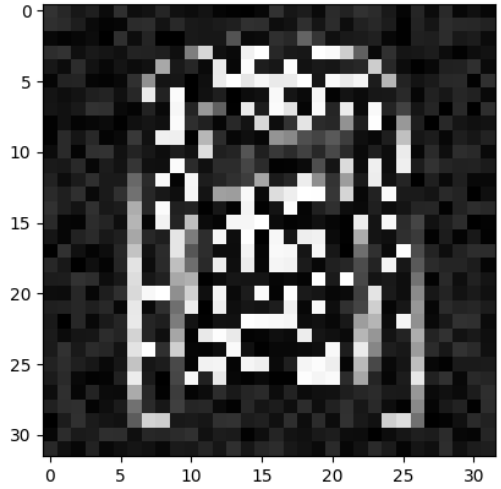

Figure 5.6: Instance of an Image from the Fashion-MNIST [57] dataset, depicting a pullover, as seen in Figure 5.4, with noise added to the image.

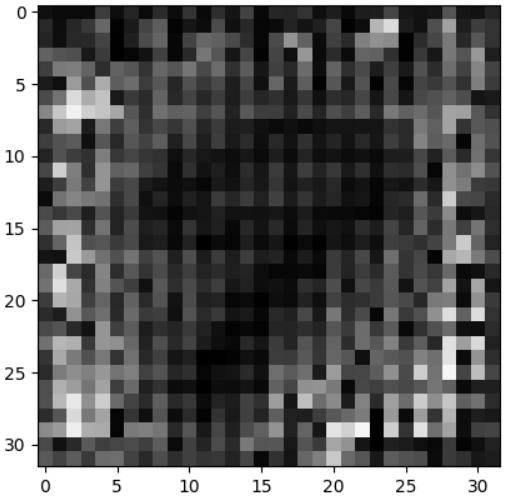

Figure 5.7: Saliency Map generated for the image as seen in Figure 5.3, taken from the MNIST [29] dataset, when it is passed through the inner U-Net segmentation model. Here, we can see the noise patterns mapped in terms of the context of the image, in this case, the number 9.

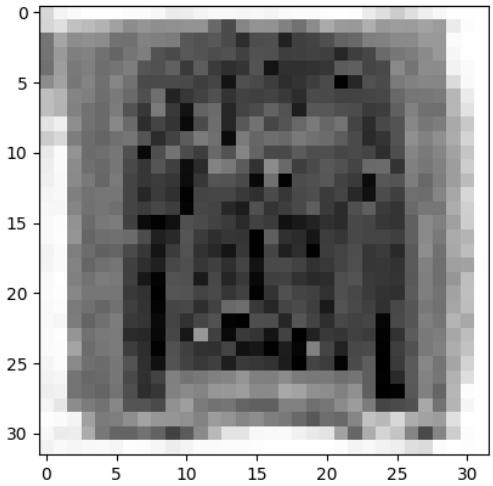

Figure 5.8: Saliency Map generated for the image as seen in Figure 5.4, taken from the Fashion-MNIST [57] dataset, when it is passed through the inner U-Net segmentation model. Here, we can see the noise patterns mapped in terms of the context of the image, in this case, a pullover.

Figures 5.3 and 5.4 depict the original images from the MNIST [29] and FashionMNIST [57] datasets. On adding noise to them, artificially, they look similar images as seen in Figures 5.5 and 5.6. Once these images with noise are passed through the proposed model, they output saliency maps that highlight the noise artifacts in the image and depict relevant structural information of the object in the image. On analyzing the saliency maps, as seen in Figures 5.7 and 5.8, we notice the structure of the object being highlighted along with the presence of noise around it. The output of the inner U-Net system changes with the amount and type of noise added to those images. This disparity is seen between Figures 5.7 and 5.8, where the two images depicted, is a result of different type of noise being added to both images from respective datasets.

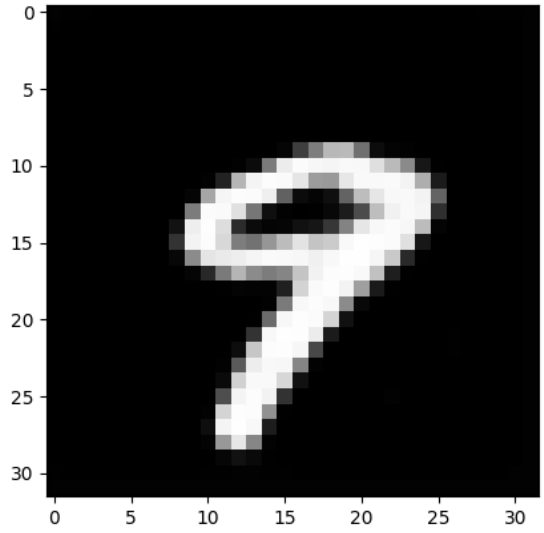

Figure 5.9: The denoised and enhanced version of the noisy Image as seen in Figure 5.5, taken from the MNIST [29] dataset, when it is passed through a proposed architecture. Here, we can see how well-defined the image context is along with the extent to which denoising was done.

The output of the inner segmentation U-Net model, Saliency Map, is essential in providing a context-aware enhancement system. If these inner models do not depict or learn the correct patterns of noise and image structure, the output deviates from the ground truth. Defining a system that can optimally recognize noise artifacts and image structure pave the way for scaling of working of the system. When the Saliency Maps generated from system trained with noise modeled as a Mixture of Gaussians, cause the proposed system to perform optimally, testing the system's effectiveness on a more complex dataset, would suggest the importance of the U-Net model for feature extraction.

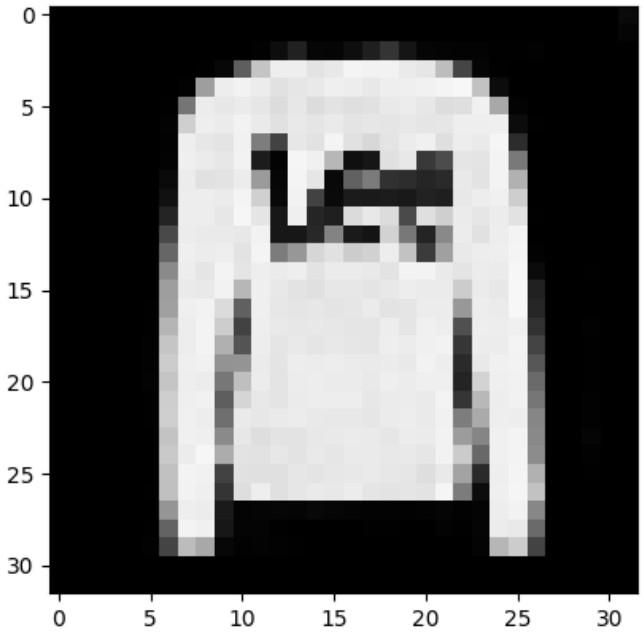

Figure 5.10: The denoised and enhanced version of the noisy Image as seen in Figure 5.6, taken from the Fashion-MNIST [57] dataset, when it is passed through a proposed architecture. Here, we can see how well-defined the image context is along with the extent to which denoising was done.

Figures 5.9 and 5.10 show the final, denoised and enhanced output of the proposed model. On the enhanced image from the MNIST dataset [29] as seen in Figure 5.9, we notice that the line created within the circular section of the number "9" in the original image has been removed to create an image that is more aesthetically pleasing and closer to what the number "9" should look. On placing them next to each other, we can notice the slight improvements brought about by the proposed model, as seen in Figure 5.11 that depicts the noisy image, denoised and enhanced image and ground truth respectively.

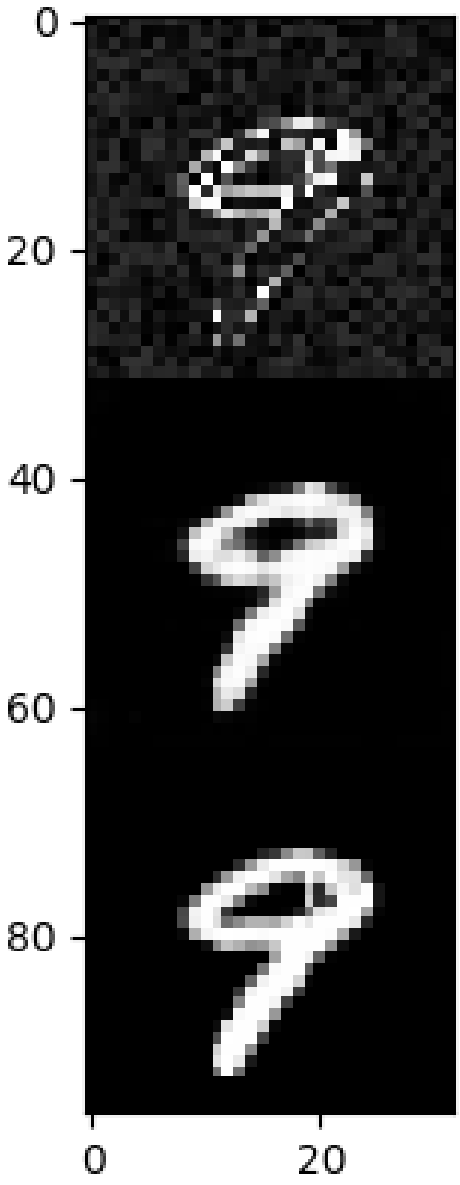

Figure 5.11: A combined version of the image from Figure 5.3, taken from the MNIST [29] dataset, that depicts the image at three different stages of the model, where the topmost image is the original noisy image, the image in between is the denoised and enhanced image and the image at the bottom is the original image before noise was added to it.

On taking a closer look at the Fashion-MNIST example, one can notice the color gradient has become smoother in the enhanced image compared to the ground truth. We also notice how the pixels around the letters "Lee" on shirt appear more blended in comparison. Subtle changes like this are the enhancements brought over by the proposed system. On placing them next to each other, one can notice these subtle improvements, as seen in Figure 5.12. The perturbations brought about in the inputs image, does not accurately depict the letters "Lee" visibly, but the saliency map generated by the U-Net model identifies the characters, as seen in the Figure 5.8.

When dealing with evaluations based on computer vision tasks, we have discussed and analyzed the results and inferred reasons for developing better and more optimized machine learning and deep learning algorithms.

We took the test set of the Fashion-MNIST dataset [57] and used these images as input to a pre-trained ResNet model [20]. This pre-trained ResNet model was trained to generalize on Fashion-MNIST's training set.

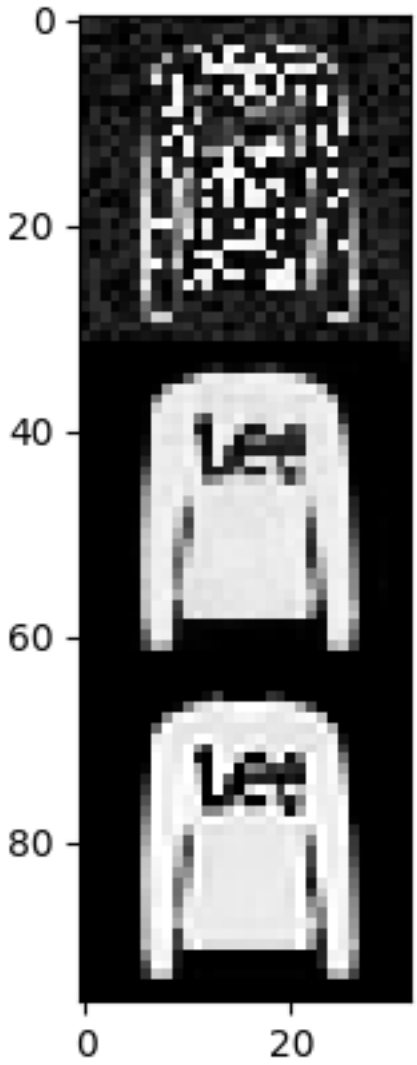

Figure 5.12: A combined version of the image from Figure 5.3, taken from the Fashion-MNIST [57] dataset, that depicts the image at three different stages of the model, where the topmost image is the original noisy image, the image in between is the denoised and enhanced image and the image at the bottom is the original image before noise was added to it.

Table 5.3: Metric scores of the effectiveness of the original test images on the pretrained model on the FashionMNIST dataset.

Class ID	Class Name	Precision	Recall	F1-score
0	**T-shirt/top**	0.890	0.870	0.880
1	**Trouser**	0.960	0.970	0.965
2	**Pullover**	0.880	0.850	0.865
3	**Dress**	0.920	0.890	0.905
4	**Coat**	0.920	0.910	0.915
5	**Sandal**	0.980	0.990	0.985
6	**Shirt**	0.850	0.810	0.830
7	**Sneaker**	0.970	0.950	0.960
8	**Bag**	0.990	0.990	0.990
9	**Ankle boot**	0.960	0.970	0.965
Average Scores		0.932	0.920	0.926

Once the model performed optimally on the validation set, we used the test set and recorded metric results for all the classes, as seen in Table 5.3. The model was able to detect and classify each of the images into their respective classes. Once this model performed optimally on the original test set, we passed all the images of the test set through our proposed architecture and re-recorded the same metrics in Table 5.4, we saw an improvement in the metric scores overall. On comparing the metric scores from both the scenarios, we notice that the average Precision, Recall, and F1-scores have improved from 0.932, 0.920, and 0.926 to 0.955, 0.947, and 0.951, respectively. We observe an improvement of 2.3%, 2.7%, and 2.5% on the average Precision, Recall, and F1-scores, respectively.

Table 5.4: Metric scores of the effectiveness of the enhanced test images on the pretrained model on the FashionMNIST dataset after being passed through the proposed system.

Class ID	Class Name	Precision	Recall	F1-score
0	**T-shirt/top**	0.890	0.880	0.885
1	**Trouser**	0.960	0.970	0.965
2	**Pullover**	0.930	0.890	0.910
3	**Dress**	0.940	0.930	0.935
4	**Coat**	0.970	0.980	0.975
5	**Sandal**	1.000	0.990	0.995
6	**Shirt**	0.920	0.910	0.915
7	**Sneaker**	0.970	0.950	0.960
8	**Bag**	0.990	0.990	0.990
9	**Ankle boot**	0.980	0.980	0.980
Average Scores		0.955	0.947	0.951

To ensure the proposed model performs optimally on unseen images, we built a system to remove as many types of noise as possible. To reach this stage, we built a noise model using a Mixture of Gaussians that would represent multiple patterns of noise. The resulting images looked similar to those seen in Figures 5.13 and 5.14.

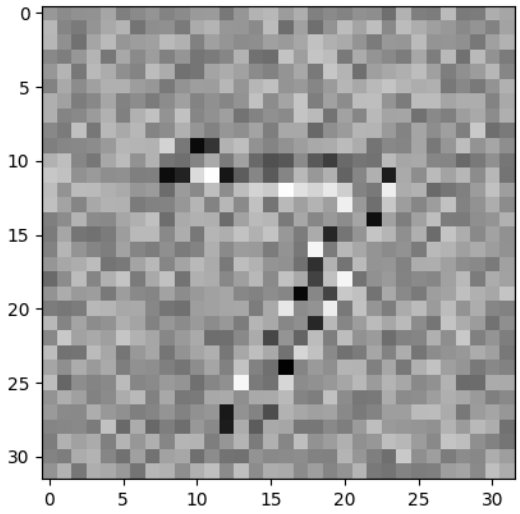

Figure 5.13: Instance of an Image from the MNIST [29] dataset, depicting the number 7, with noise modeled as a Mixture of Gaussians is added to the image.

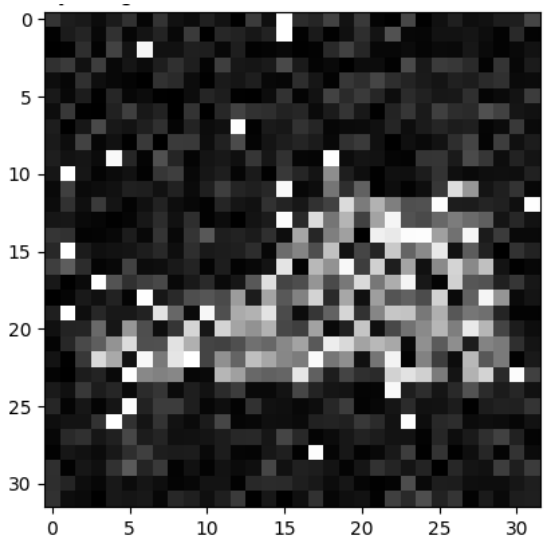

Figure 5.14: Instance of an Image from the Fashion-MNIST [57] dataset, depicting a shoe, with noise modeled as a Mixture of Gaussians is added to the image.

To test the effectiveness of this blind denoising feature, we tested the system on images with unseen noise added to them. Figures 5.15 and 5.16 depict the original image and Saliency Map generated, and a stacked image of noisy, denoised and enhanced and original images when the model was tested on images with unseen noise (a combination of Speckle and Gaussian noise) added to it. From Figure 5.16, we can notice the finesse with which the model denoised the image.

On running the same test on the Fashion-MNIST dataset, the model's effectiveness was similar to that when testing MNIST data. Figures 5.17 and 5.18 depict the original image and Saliency Map generated, and a stacked image of noisy, denoised and enhanced and original images when the model was tested on images with unseen noise (a combination of Speckle and Gaussian noise) added to it. During the process of denoising the Fashion-MNIST image [57] with unseen noise added to it, the Saliency Map, as seen in Figure 5.17, we notice how the pattern of noise depicted in terms of the context of the image, in this case the shape of the shoe.

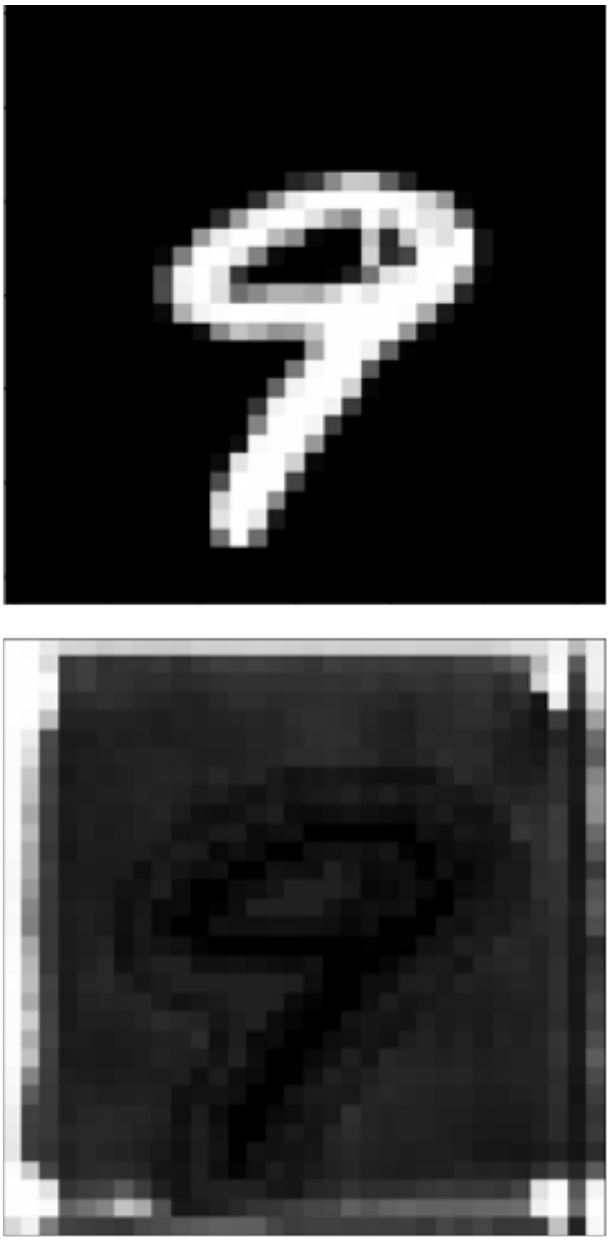

Figure 5.15: The original image and the corresponding Saliency Map generated from the inner segmentation U-Net model, taken from the MNIST dataset [29] to which we added a combination Speckle and Gaussian noise.

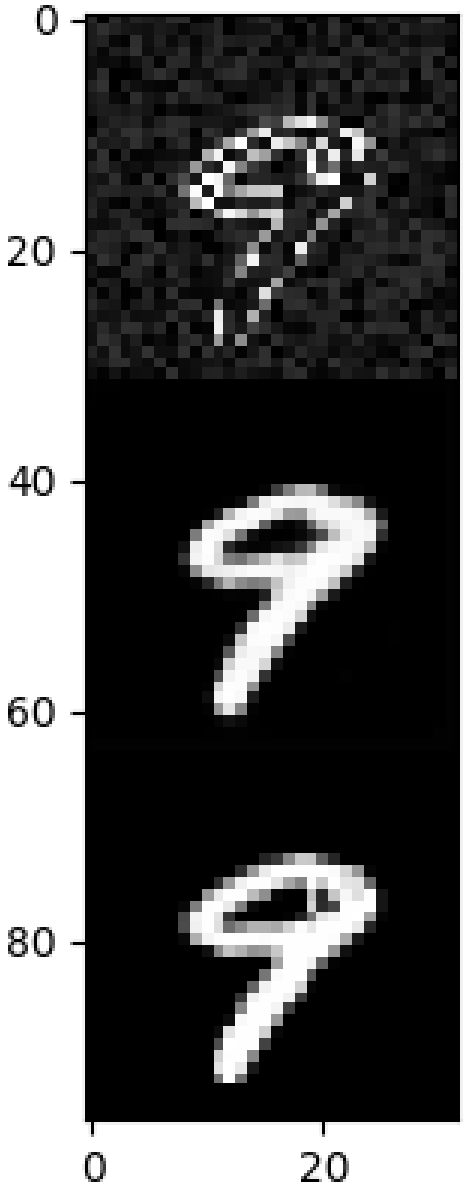

Figure 5.16: An image with the noisy image, denoised and enhanced image and original image stacked in that order for a clear representation of the image during the process, when a combination Speckle and Gaussian noise was added to an image from the MNIST dataset [29].

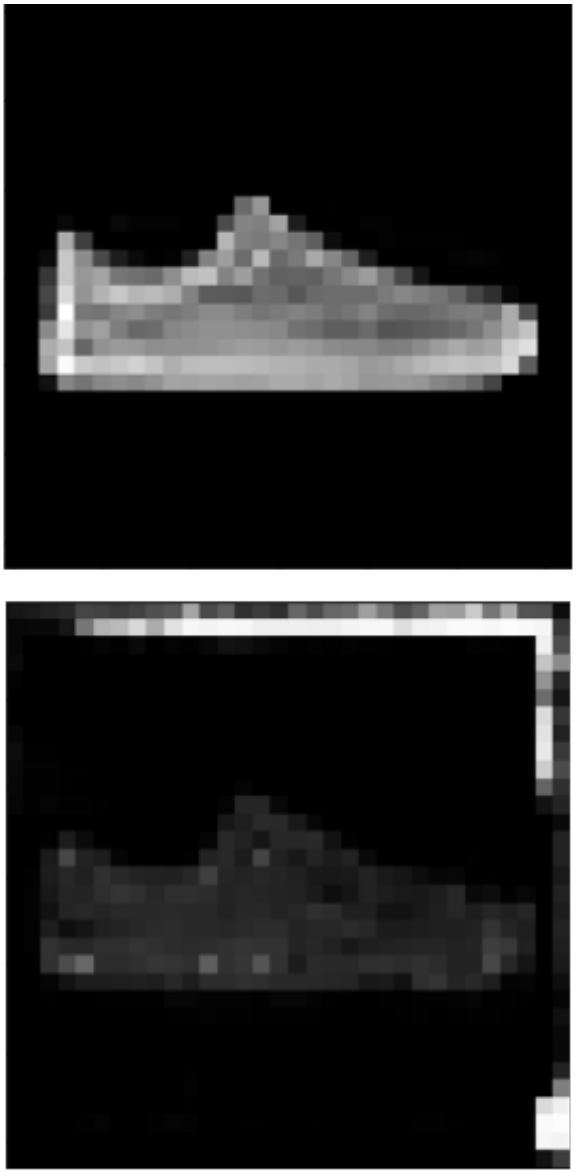

Figure 5.17: The original image and the corresponding Saliency Map generated from the inner segmentation U-Net model, taken from the Fashion-MNIST dataset [57] to which we added a combination Speckle and Gaussian noise.

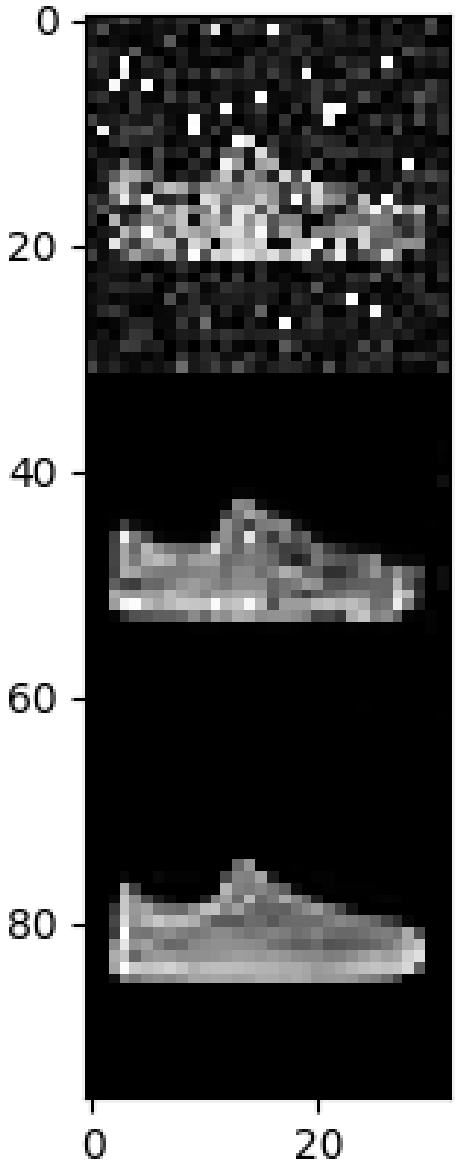

Figure 5.18: An image with the noisy image, denoised and enhanced image and original image stacked in that order for a clear representation of the image during the process, when a combination Speckle and Gaussian noise was added to an image from the Fashion-MNIST dataset [57].

Table 5.5: PSNR and SSIM metric scores for models trained with noise modeled as a Mixture of Gaussians on the MNIST and Fashion-MNIST datasets respectively, when tested on images from the ImageNet dataset.

Trained on Dataset	**Metric**	**Noisy Images**	**Denoised Images**
MNIST	**PSNR**	13.7270	19.5170
	SSIM	0.2175	0.8840
Fashion-MNIST	**PSNR**	12.1119	18.5690
	SSIM	0.2130	0.6890

To test the system for improvements in metrics on images with unseen noise artifacts, we test the model that was trained on the MNIST and Fashion-MNIST datasets using noise modeled as a Mixture of Gaussians (MoG). Any improvements in metric scores, in this scenario, correlates to noise modeled as a Mixture of Gaussians (MoG) to depict the generic occurrence of noise in images correctly.

Once we got models that performed optimally, we tested these models on a larger and complex dataset, ImageNet [8]. On testing the effectiveness of our system trained on the MNIST and Fashion-MNIST dataset with noise modeled as a Mixture of Gaussians. Using ImageNet dataset's validation set as our test data, we got results that showed significant improvements over systems built over both datasets, MNIST and Fashion-MNIST. Improvements in PSNR and SSIM metric scores can be seen in Table 5.5. The PSNR and SSIM scores calculated on the system trained with the MNIST dataset have better results, 19.5170 and 0.8840, respectively than with the system trained with the Fashion-MNIST dataset.

We believe that the added complexity introduced by Fashion-MNIST compared to MNIST was the reason for the relatively better performance. From these tests, we can infer a much better performance on unseen images if the system trains from first principles using the ImageNet dataset. The added complexity and high amount of data points would help the model generalize more optimally.

Until this point, we have only discussed the effect of noise affecting images visibly. Szegedy et al. [51] and Nguyen et al. [40] discuss how a small amount of noise perturbations, when added to images, does not affect the visual structure of the image, but this small amount of perturbations are enough to confuse complex deep neural networks. We utilized one such perturbation, FGSM (Fast Gradient Sign Method). Linearity, introduced in the inner layers of the model and high dimensionality of input features, is enough to generate this type of noise in images. Being able to perform context-aware denoising in such situations is of high importance, as this type of noise introduces very little to no changes in images visible to the human eye.

Goodfellow et al. [18] defines FGSM as:

$$adv_x = x + \epsilon * sign(\nabla_x \cdot J(\theta, x, y)) \tag{5.11}$$

Where,

- adv_x represents the Adversarial image

- x represents the original input image
- y represents the original input label
- ϵ represents the multiplier that ensures that the perturbations are small
- θ represents model parameters
- J represents the loss

This type of noise occurs when the ϵ values are very low. To test the effectiveness of our proposed system, we tested the system against multiple values of ϵ ranging from 0 to 0.3. where extreme visible changes are observed around 0.3. We made use of a pre-trained "LeNet" model, defined by LeCunn et al. [29], that classified the digits in the following two scenarios:

1. Original Images affected with Adversarial (FGSM) noise
2. Original Images affected with Adversarial (FGSM) noise after being passed through proposed model, pretrained with noise represented as a mixture of Gaussians.

To test the Fashion-MNIST dataset being affected by adversarial noise, we used a pre-trained ResNet model, fine-tuned to fit the Fashion-MNIST data distribution and tested similar to the process used with MNIST [29].

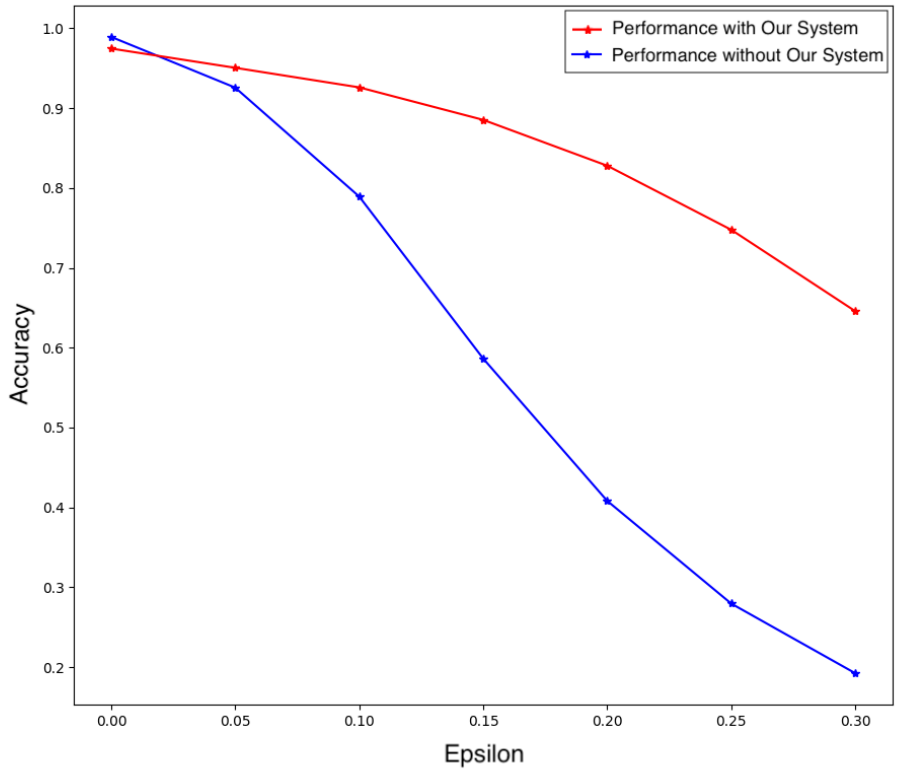

Figure 5.19: The Accuracy-Epsilon plot for values of epsilon ranging from 0 to 0.3, where the accuracy is of a classification model on the MNIST dataset [29].

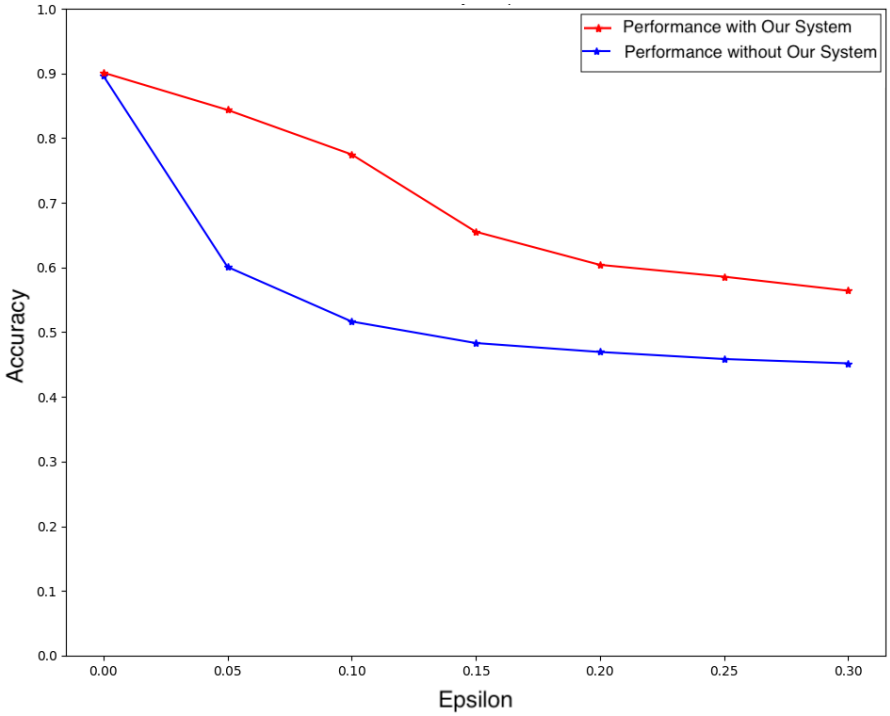

Figure 5.20: The Accuracy-Epsilon plot for values of epsilon ranging from 0 to 0.3, where the accuracy is of a classification model on the Fashion-MNIST dataset [57].

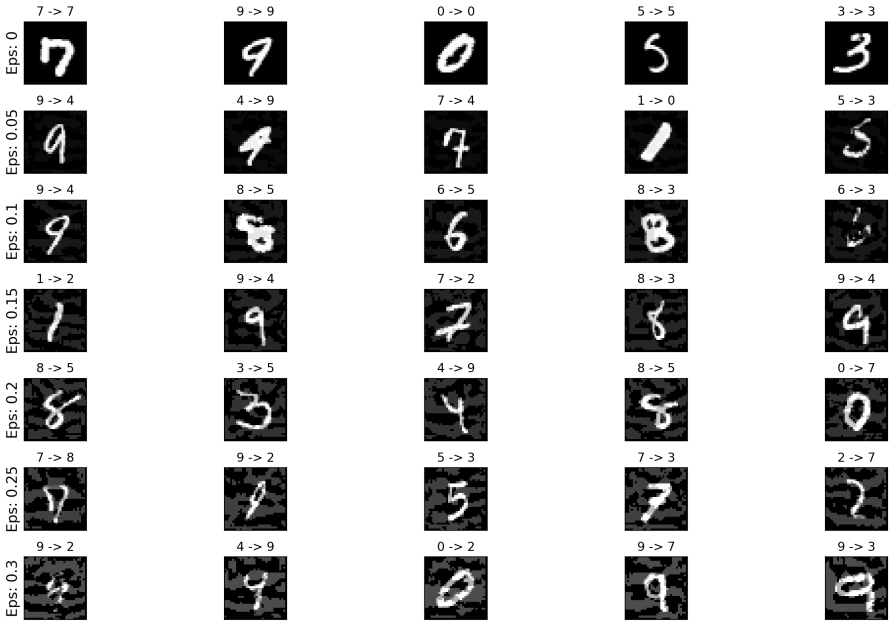

Figure 5.21: Examples of images from the test set that were affected by FGSM noise and classified by a pretrained LeNet model. This image depicts the image being tested, with its ground truth and predicted values displayed above the image, in that order respectively. The images were taken from the MNIST dataset [29].

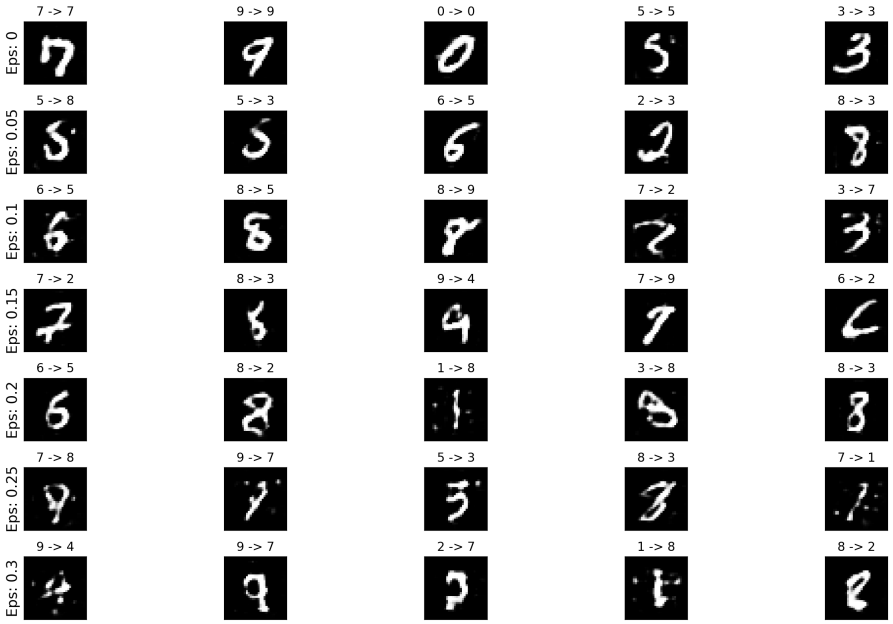

Figure 5.22: Examples of images from the test set that were affected by FGSM noise, passed through the proposed system, and then classified by a pretrained LeNet model. This image depicts the image being tested, with its ground truth and predicted values displayed above the image, in that order respectively. The images were taken from the MNIST dataset [29].

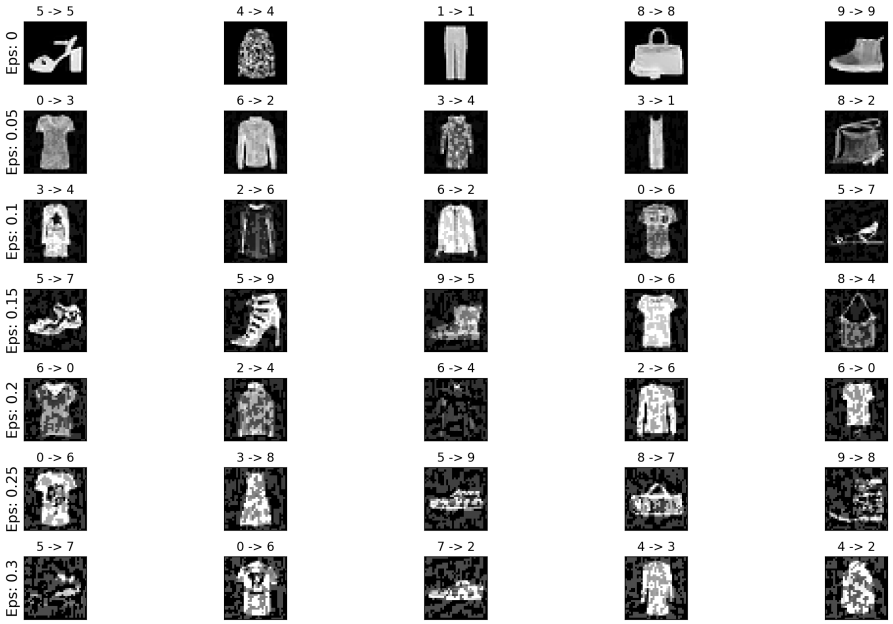

Figure 5.23: Examples of images from the test set that were affected by FGSM noise and classified by a pretrained ResNet model. This image depicts the image being tested, with its ground truth and predicted values displayed above the image, in that order respectively. The images were taken from the Fashion-MNIST dataset [57].

Figure 5.24: Examples of images from the test set that were affected by FGSM noise, passed through the proposed system, and then classified by a pretrained ResNet model. This image depicts the image being tested, with its ground truth and predicted values displayed above the image, in that order respectively. The images were taken from the Fashion-MNIST dataset [57].

In both the plots, 5.19 and 5.20, there is a significant improvement in the classification accuracy after the noisy images are denoising and enhanced using our proposed system. The difference in performance accuracy is much higher in MNIST in comparison, and we can infer from Figure 5.19 that at an epsilon value of 0.3, the classification performance drops significantly to 20% versus around 65% in the other scenario. A few examples are depicted in Figures 5.21 and 5.22 that are taken from MNIST and 5.23 and 5.24 that are taken from Fashion-MNIST. These depict images with their ground truth

and predicted class mentioned above the image. We can see how numbers that are easily identifiable by us humans are misclassified with the smallest epsilon value.

Figures 5.25 and 5.26 depict images from MNIST and Fashion-MNIST respectively, where the topmost image is the one affected with the perturbations, the middle one is the denoised and enhanced version of the image found as an output of the proposed model, and the bottom-most image is the original image. One can notice how there seem to be deceivingly no visual changes in the image affected by noise, and its similarity to the original image. As the epsilon value is increased in Equation 5.11, the effect can be noticed visually.

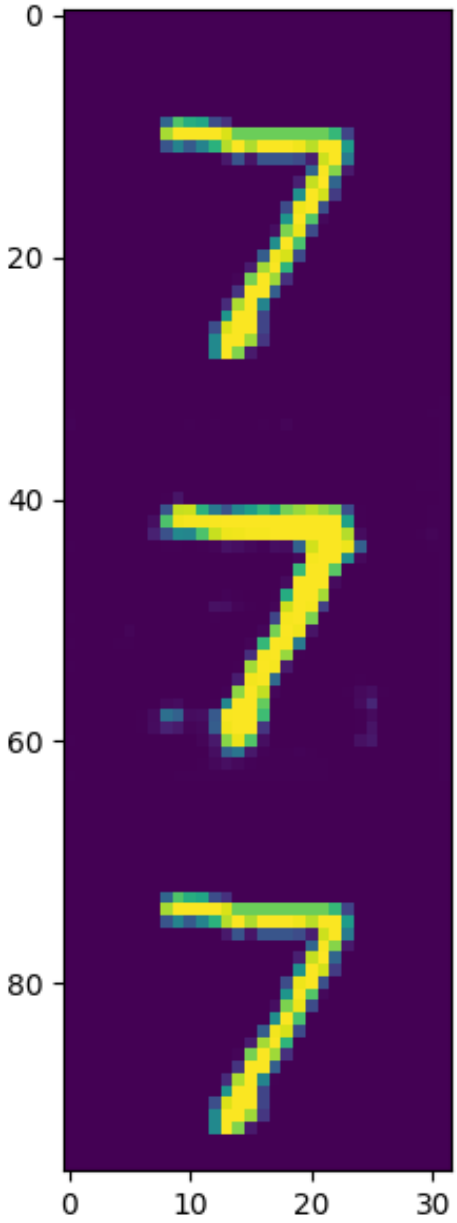

Figure 5.25: An image with the noisy image, denoised and enhanced image, and original image stacked in that order for a clear representation of the image during the process of denoising, when a Adversarial (FGSM) perturbations was added to the image, taken from the MNIST dataset [29].

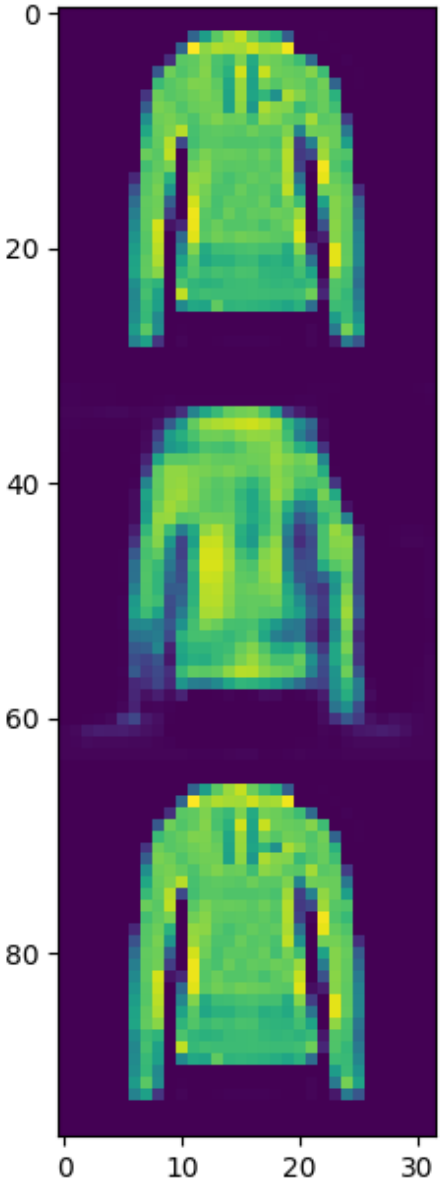

Figure 5.26: An image with the noisy image, denoised and enhanced image, and original image stacked in that order for a clear representation of the image during the process of denoising, when a Adversarial (FGSM) perturbations was added to the image, taken from the Fashion-MNIST dataset [57].

Chapter 6

Future Work

Given the time and scope of this thesis, we were able to run experiments with most of the hyperparameters set initially. An extension to this study would be to look into each model's efficacy, by fine-tuning each models' hyperparameters. Apart from fine-tuning the hyperparameters, the vanilla convolutional layers in the encoding stage could be replaced by more complex and deeper networks, such as those found in the feature extraction layers of the ResNet model. We could also look into replacing the outer model's decoding layers from CNN-based to LSTM-based layers. Changes in the outer model's decoding structure can be made by introducing the LSTM model and the self-attention mechanism. This could improve the reconstruction process by generating a finer content-aware denoised and enhanced image.

Improvements in representing the multiple noise patterns, as a mixture of Gaussians, would help build a system that would perform blind denoising and enhancing. Ideally, we aimed to build a blind denoising system, blind because this system could ideally denoise all possible types of noise artifacts present in images.

One could also look into image size-independent denoising and enhanc-

ing the system. Ideally, not all image datasets have the same image size. Converting every image to a standard size, followed by performing an image to image conversion (noisy to clean) and then reconverting it back to its original size, would be an essential goal to accomplish. While working on the denoising process in this case, one would also have to account for noise generated during the upsampling and downsampling process. Only then will the system indeed be size-independent.

To build a scalable denoising system capable of denoising and enhancing multiple types of images, training the proposed architecture on a large and complex dataset like ImageNet from first principles might prove more effective than transfer learning from models built on MNIST and Fashion-MNIST datasets.

In the process of this study, we have only scratched the surface with adversarial noise. Dealing with adversarial noise and deep fake images, generated by masking and mapping features of one image on the other, can make this problem extremely difficult to use. With every image, one would have to check for the presence of adversarial noise and if the image appears to be a deep fake. Due to the study's limitation of time and scope, we could not delve deeper into this aspect of the study.

Chapter 7

Conclusion

In this thesis we have introduced two image denoising architectures, one that only uses the output of the inner U-Net segmentation model, the Saliency Map, and the other takes the advantage of the Saliency Map along with the original noisy image. We have observed an increase in effectiveness in denoising when the image is reconstructed by stacking the Saliency Map and the original noisy image compared to other proposed architecture and the baseline. The use of skip connections in deep architectures is an important structural element that helps alleviate the vanishing gradients problem during backpropagation. PSNR scores improved by 7.814 and 9.5331, and SSIM scores improved by 0.0258 and 0.1544 for the MNIST and Fashion-MNIST datasets respectively, when skip connections were introduced in the architecture.

The effectiveness of the models trained with L_2 loss was more consistent over our proposed architectures, hyperparameters, and all other experiments compared to all other loss functions. ElasticNet, as defined by Equation 2.15 with a λ of 0.75, had the best results compared to all other loss functions but did not produce consistent results. Using Adam optimizer helped speed up the convergence process in comparison to Stochastic Gradient Descent (SGD).

Consequently, the resulting systems improved the effectiveness on the computer vision task of object detection and classification, where Precision, Recall, and F1-scores improved by 2.3%, 2.7%, and 2.5%, respectively.

Due to the presence of multiple patterns of noise in a variety of images, modeling a system with these multiple noise types makes the process non-trivial. We empirically represent noise as a mixture of Gaussians. Making use of this noise representation enables our proposed system to perform optimally in various scenarios. On optimizing our systems' parameters, we tested our system on a more complex and larger dataset, ImageNet [8]. Training our system on the ImageNet dataset would be a step towards building a completely blind denoising and context-aware image enhancement system.

While dealing with a variety of noise types that visually affect images, there are adversarial perturbations that affect images structurally but make minimal changes in the image, aesthetically and visually. Working with images that have these perturbations can get extremely complicated as they can easily *fool* complex deep learning systems. Our system successfully mitigated this noise by improving the effectiveness of the model by 46% on the MNIST dataset and 15% on the Fashion-MNIST dataset when epsilon was 0.3, which is considered high for adversarial noise.

Bibliography

[1] Hina Ajmal, Saad Rehman, Umar Farooq, Qurrat U Ain, Farhan Riaz, and Ali Hassan. Convolutional neural network based image segmentation: a review. In *Pattern Recognition and Tracking XXIX*, volume 10649, page 106490N. International Society for Optics and Photonics, 2018.

[2] Marco Bevilacqua, Aline Roumy, Christine Guillemot, and Marie Line Alberi-Morel. Low-complexity single-image super-resolution based on nonnegative neighbor embedding. 2012.

[3] Ajay Kumar Boyat and Brijendra Kumar Joshi. A review paper: Noise models in digital image processing. *Signal & Image Processing*, 6(2):63, 2015.

[4] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[5] Tim Brooks, Ben Mildenhall, Tianfan Xue, Jiawen Chen, Dillon Sharlet, and Jonathan T Barron. Unprocessing images for learned raw denoising. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11036–11045, 2019.

[6] Adam Byerly, Tatiana Kalganova, and Ian Dear. A branching and merging convolutional network with homogeneous filter capsules. *arXiv*, pages

arXiv–2001, 2020.

[7] Bo Dai, Sanja Fidler, Raquel Urtasun, and Dahua Lin. Towards diverse and natural image descriptions via a conditional gan. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2970–2979, 2017.

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[9] Yubin Deng, Chen Change Loy, and Xiaoou Tang. Image aesthetic assessment: An experimental survey. *IEEE Signal Processing Magazine*, 34(4):80–106, 2017.

[10] Yubin Deng, Chen Change Loy, and Xiaoou Tang. Aesthetic-driven image enhancement by adversarial learning. In *2018 ACM Multimedia Conference on Multimedia Conference*, pages 870–878. ACM, 2018.

[11] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Learning a deep convolutional network for image super-resolution. In *European conference on computer vision*, pages 184–199. Springer, 2014.

[12] Michal Drozdzal, Eugene Vorontsov, Gabriel Chartrand, Samuel Kadoury, and Chris Pal. The importance of skip connections in biomedical image

segmentation. In *Deep Learning and Data Labeling for Medical Applications*, pages 179–187. Springer, 2016.

[13] Azuma Fujimoto, Toru Ogawa, Kazuyoshi Yamamoto, Yusuke Matsui, Toshihiko Yamasaki, and Kiyoharu Aizawa. Manga109 dataset and creation of metadata. In *Proceedings of the 1st International Workshop on coMics ANalysis, Processing and Understanding*, page 2. ACM, 2016.

[14] Sahaj Gandhi, Behrooz Mansouri, Ricardo Campos, and Adam Jatowt. Event-related query classification with deep neural networks. In *Companion Proceedings of the Web Conference 2020*, pages 324–330, 2020.

[15] Lovedeep Gondara. Medical image denoising using convolutional denoising autoencoders. In *2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW)*, pages 241–246. IEEE, 2016.

[16] Rafael González and Richard Woods. Digital image processing. isbn: 9780131687288. *Prentice Hall*, 60, 2008.

[17] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[18] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples, 2014.

[19] Kazi Nazmul Haque, Mohammad Abu Yousuf, and Rajib Rana. Image denoising and restoration with cnn-lstm encoder decoder with direct attention. *arXiv preprint arXiv:1801.05141*, 2018.

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[21] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[22] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja. Single image super-resolution from transformed self-exemplars. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5197–5206, 2015.

[23] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[24] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001–.

[25] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[26] Shu Kong, Xiaohui Shen, Zhe Lin, Radomir Mech, and Charless Fowlkes. Photo aesthetics ranking network with attributes and content adaptation. In *European Conference on Computer Vision*, pages 662–679. Springer, 2016.

[27] M Kuwahara, K Hachimura, S Eiho, and M Kinoshita. Processing of ri-angiocardiographic images. In *Digital processing of biomedical images*, pages 187–202. Springer, 1976.

[28] Andréia Seixas Leal and Henrique Mohallem Paiva. A new wavelet family for speckle noise reduction in medical ultrasound images. *Measurement*, 140:572–581, 2019.

[29] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[30] Tao Lin and Salah Bourennane. Survey of hyperspectral image denoising methods based on tensor decompositions. *EURASIP journal on Advances in Signal Processing*, 2013(1):186, 2013.

[31] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[32] Alexander Loui, Jiebo Luo, Shih-Fu Chang, Dan Ellis, Wei Jiang, Lyndon Kennedy, Keansub Lee, and Akira Yanagawa. Kodak's consumer video benchmark data set: Concept definition and annotation. In *Proceedings of the International Workshop on Workshop on Multimedia Information Retrieval*, MIR '07, page 245–254, New York, NY, USA, 2007. Association for Computing Machinery.

[33] Xiaojiao Mao, Chunhua Shen, and Yu-Bin Yang. Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections. In *Advances in neural information processing systems*, pages 2802–2810, 2016.

[34] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, and Zhen Wang. Multi-class generative adversarial networks with the l2 loss function. *arXiv preprint arXiv:1611.04076*, 5, 2016.

[35] Sébastien Marcel and Yann Rodriguez. Torchvision the machine-vision package of torch. In *Proceedings of the 18th ACM international conference on Multimedia*, pages 1485–1488. ACM, 2010.

[36] David Martin, Charless Fowlkes, Doron Tal, Jitendra Malik, et al. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. Iccv Vancouver:, 2001.

[37] J Mohan, V Krishnaveni, and Yanhui Guo. A survey on the magnetic

resonance image denoising methods. *Biomedical signal processing and control*, 9:56–69, 2014.

[38] Naila Murray, Luca Marchesotti, and Florent Perronnin. Ava: A large-scale database for aesthetic visual analysis. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2408–2415. IEEE, 2012.

[39] Makoto Nagao and Takashi Matsuyama. Edge preserving smoothing. *Computer graphics and image processing*, 9(4):394–407, 1979.

[40] Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 427–436, 2015.

[41] A Emin Orhan and Xaq Pitkow. Skip connections eliminate singularities. *arXiv preprint arXiv:1701.09175*, 2017.

[42] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett,

editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[43] Pietro Perona and Jitendra Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on pattern analysis and machine intelligence*, 12(7):629–639, 1990.

[44] Tobias Plotz and Stefan Roth. Benchmarking denoising algorithms with real photographs. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[45] NK Ragesh, AR Anil, and R Rajesh. Digital image denoising in medical ultrasound images: a survey. In *Icgst Aiml-11 Conference, Dubai, UAE*, volume 12, page 14, 2011.

[46] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[47] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[48] Rupesh K Srivastava, Klaus Greff, and Jürgen Schmidhuber. Training very deep networks. In *Advances in neural information processing systems*, pages 2377–2385, 2015.

[49] Suresh Sudha, GR Suresh, and R Sukanesh. Speckle noise reduction in ultrasound images by wavelet thresholding based on weighted variance. *International journal of computer theory and engineering*, 1(1):7, 2009.

[50] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[51] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks, 2013.

[52] Hossein Talebi and Peyman Milanfar. Nima: Neural image assessment. *IEEE Transactions on Image Processing*, 27(8):3998–4011, 2018.

[53] Chunwei Tian, Yong Xu, Lunke Fei, and Ke Yan. Deep learning for image denoising: a survey. In *International Conference on Genetic and Evolutionary Computing*, pages 563–572. Springer, 2018.

[54] Tong Tong, Gen Li, Xiejie Liu, and Qinquan Gao. Image super-resolution using dense skip connections. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4799–4807, 2017.

[55] Dimpy Varshni, Kartik Thakral, Lucky Agarwal, Rahul Nijhawan, and Ankush Mittal. Pneumonia detection using cnn based feature extrac-

tion. In *2019 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT)*, pages 1–7. IEEE, 2019.

[56] Zhou Wang, Alan C Bovik, Hamid R Sheikh, Eero P Simoncelli, et al. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[57] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[58] Xin Yi, Ekta Walia, and Paul Babyn. Generative adversarial network in medical imaging: A review. *Medical image analysis*, page 101552, 2019.

[59] Changqian Yu, Jingbo Wang, Changxin Gao, Gang Yu, Chunhua Shen, and Nong Sang. Context prior for scene segmentation. *arXiv preprint arXiv:2004.01547*, 2020.

[60] Tiantian Yuan, Shagan Sah, Tejaswini Ananthanarayana, Chi Zhang, Aneesh Bhat, Sahaj Gandhi, and Raymond Ptucha. Large scale sign language interpretation. In *2019 14th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2019)*, pages 1–5. IEEE, 2019.

[61] Roman Zeyde, Michael Elad, and Matan Protter. On single image scale-up using sparse-representations. In *International conference on curves and surfaces*, pages 711–730. Springer, 2010.

[62] Hang Zhao, Orazio Gallo, Iuri Frosio, and Jan Kautz. Loss functions for image restoration with neural networks. *IEEE Transactions on computational imaging*, 3(1):47–57, 2016.